REPORT DOCUMENTATION PAGE

2

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
	Technical Report	-

4. TITLE AND SUBTITLE

Senior Honors Thesis: Load Balancing in Stochastic Networks: Algorithms, Analysis, and Game Theory

5a. CONTRACT NUMBER
W911NF-12-1-0222

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS

Katrina Kardassakis

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES

Brown University
Box 1929

Providence, RI 02912 -9093

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
61759-MA.15

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

The classic randomized load balancing model is the so-called supermarket model, which describes a system in which customers arrive to a service center with n parallel servers according to a Poisson process with rate ?n, where ? < 1. Upon arrival, each customer samples d queues independently and uniformly at random before joining the shortest of those sampled. Customers are served according to a first-in first-out (FIFO) scheduling rule, and their service times are assumed to be mutually independent and exponentially distributed with unit mean ? = 1. Any ties that may occur are broken randomly. When d = 1, the model reduces to a system of n independent M/M/1 queues

15. SUBJECT TERMS

mean-field limits, supermarket model, thresholds, game, randomized load balancing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Paul Dupuis
UU	UU	UU	UU		19b. TELEPHONE NUMBER 401-863-3238

Report Title

Senior Honors Thesis: Load Balancing in Stochastic Networks: Algorithms, Analysis, and Game Theory

ABSTRACT

The classic randomized load balancing model is the so-called supermarket model, which describes a system in which customers arrive to a service center with n parallel servers according to a Poisson process with rate ?n, where ? < 1. Upon arrival, each customer samples d queues independently and uniformly at random before joining the shortest of those sampled. Customers are served according to a first-in first-out (FIFO) scheduling rule, and their service times are assumed to be mutually independent and exponentially distributed with unit mean ? = 1. Any ties that may occur are broken randomly. When d = 1, the model reduces to a system of n independent M/M/1 queues, for which it is a classical result that the stationary queue length distribution at a single queue is geometric with parameter ?, and thus has an exponential decay rate. When d ? 2, the model is not exactly solvable, but asymptotic results show that as n, the number of servers, goes to infinity, the limiting stationary distribution of a queue decays superexponentially. Moreover, the majority of this gain in performance is already obtained when d = 2. In particular, this shows that with just a slight increase in sampling cost, from d = 1 to d = 2, the performance is almost as good as in the case when all queues are sampled (that is, the Join-the-Shortest-Queue system where d = n). This phenomenon is referred to as the "power of two choices," and this classic model is well studied.

With an aim to further examine tradeoffs between performance and sampling complexity, we examine a variation of this model, which we refer to as the threshold supermarket model. In this model, upon arrival, each customer samples a single queue. If this initial queue has length greater than or equal to a certain threshold T, the customer samples one additional queue and joins the shorter of the two (with ties being broken uniformly at random). On the other hand, if the initial queue has length strictly below this threshold T, the customer simply joins the queue. We use a combination of tools from the theory of Markov jump processes, ordinary differential equations, and recursive equations to analyze the system with arbitrary threshold and quantify the tradeoffs between the cost of sampling and the performance of the system. In particular, we identify the socially optimal threshold T? that minimizes a suitable cost function that includes a marginal cost of sampling c for each customer. We also provide simulations that demonstrate the accuracy of our model's predictions. We find that when the cost of sampling is greater than zero, implementing a threshold policy to determine the number of queues to sample upon arrival significantly reduces the total cost to the system. In fact, for a given ? and marginal cost of sampling, the threshold supermarket model always outperforms the corresponding supermarket model with zero threshold. Finally, we also consider a game associated with the threshold supermarket model. In this game, each customer chooses a threshold to minimize her total cost, when the marginal sampling cost is c'. For a given "actual" marginal cost of sampling c, we identify a corresponding "imposed" marginal cost of sampling c' for the game such that the social optimum for c coincides with the Nash equilibrium for c'. The difference c' ? c can be viewed as a tax (or subsidy) that a system operator levies on individuals so that when each customer behaves selfishly to minimize her cost, the system converges to the social optimum corresponding to the marginal sampling cost cs. We discuss the very interesting implications of these results for the design of many load balancing systems.

Load Balancing in Stochastic Networks: Algorithms, Analysis, and Game Theory

Katrina Kardassakis

April 16, 2014

Undergraduate Honors Thesis
Brown University

Primary Advisor: Professor Kavita Ramanan
Secondary Advisor: Professor Pedro Dal Bo
Ph.D. Candidate Mentor: Mohammadreza Aghajani

Abstract

The classic randomized load balancing model is the so-called supermarket model, which describes a system in which customers arrive to a service center with n parallel servers according to a Poisson process with rate λn, where $\lambda < 1$. Upon arrival, each customer samples d queues independently and uniformly at random before joining the shortest of those sampled. Customers are served according to a first-in first-out (FIFO) scheduling rule, and their service times are assumed to be mutually independent and exponentially distributed with unit mean $\mu = 1$. Any ties that may occur are broken randomly. When $d = 1$, the model reduces to a system of n independent M/M/1 queues, for which it is a classical result that the stationary queue length distribution at a single queue is geometric with parameter λ, and thus has an exponential decay rate. When $d \geq 2$, the model is not exactly solvable, but asymptotic results show that as n, the number of servers, goes to infinity, the limiting stationary distribution of a queue decays superexponentially. Moreover, the majority of this gain in performance is already obtained when $d = 2$. In particular, this shows that with just a slight increase in sampling cost, from $d = 1$ to $d = 2$, the performance is almost as good as in the case when all queues are sampled (that is, the Join-the-Shortest-Queue system where $d = n$). This phenomenon is referred to as the "power of two choices," and this classic model is well studied.

With an aim to further examine tradeoffs between performance and sampling complexity, we examine a variation of this model, which we refer to as the threshold supermarket model. In this model, upon arrival, each customer samples a single queue. If this initial queue has length greater than or equal to a certain threshold T, the customer samples one additional queue and joins the shorter of the two (with ties being broken uniformly at random). On the other hand, if the initial queue has length strictly below this threshold T, the customer simply joins the queue. We use a combination of tools from the theory of Markov jump processes, ordinary differential equations, and recursive equations to analyze the system with arbitrary threshold and quantify the tradeoffs between the cost of sampling and the performance of the system. In particular, we identify the socially optimal threshold T^* that minimizes a suitable cost function that includes a marginal cost of sampling c_s for each customer. We also provide simulations that demonstrate the accuracy of our model's predictions. We find that when the cost of sampling is greater than zero, implementing a threshold policy to determine the number of queues to sample upon arrival significantly reduces the total cost to the system. In fact, for a given λ and marginal cost of sampling, the threshold supermarket model always outperforms the corresponding supermarket model with zero threshold.

Finally, we also consider a game associated with the threshold supermarket model. In this game, each customer chooses a threshold to minimize her total cost, when the marginal sampling cost is $\widehat{c_s}$. For a given "actual" marginal cost of sampling c_s, we identify a corresponding "imposed" marginal cost of sampling $\widehat{c_s}$ for the game such that the social optimum for c_s coincides with the Nash equilibrium for $\widehat{c_s}$. The difference $\widehat{c_s} - c_s$ can be viewed as a tax (or subsidy) that a system operator levies on individuals so that when each customer behaves selfishly to minimize her cost, the system converges to the social optimum corresponding to the marginal sampling cost c_s. We discuss the very interesting implications of these results for the design of many load balancing systems.

Contents

1 Introduction

A common challenge for a variety of applications in computer science and engineering is to devise simple algorithms that efficiently balance load across multiple resources so as to maximize performance. In the simplest setting, consider a single stream of customers arriving to a system of n parallel queues with independent identical servers. The question is how to route customers to queues so as to achieve a small expected queue length in steady state. One load balancing algorithm that has been proposed is the so-called Join-the-Shortest-Queue (JSQ), in which an arriving customer joins the queue with the least number of packets (with ties broken uniformly at random). When the system is stable and the service distribution is exponential, this algorithm achieves very good performance. However, when n is large, this algorithm can be infeasible or prohibitively expensive to implement as it requires knowledge of states of n queues every time a customer arrives to the system.

An alternative that has been proposed is the so-called supermarket model. In this model, customers arrive to a service center with n parallel servers according to a Poisson process with rate λn, where $\lambda < 1$, sample d of the servers independently and uniformly at random (with replacement), and wait an unknown amount of time in the shortest of the d queues sampled until served. Customers are served according to a first-in first-out (FIFO) service discipline, and their service times are assumed independent and exponentially distributed with mean $\mu = 1$. Any ties that may occur are broken randomly. Such systems are ubiquitous and can be found in a variety of situations, including cloud computing, hashing, data centers, resource allocation, and other service specific systems such as banks and grocery stores.

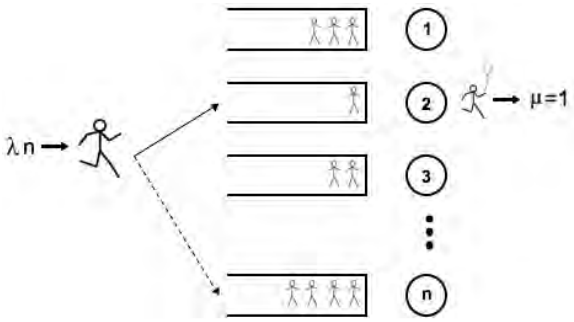
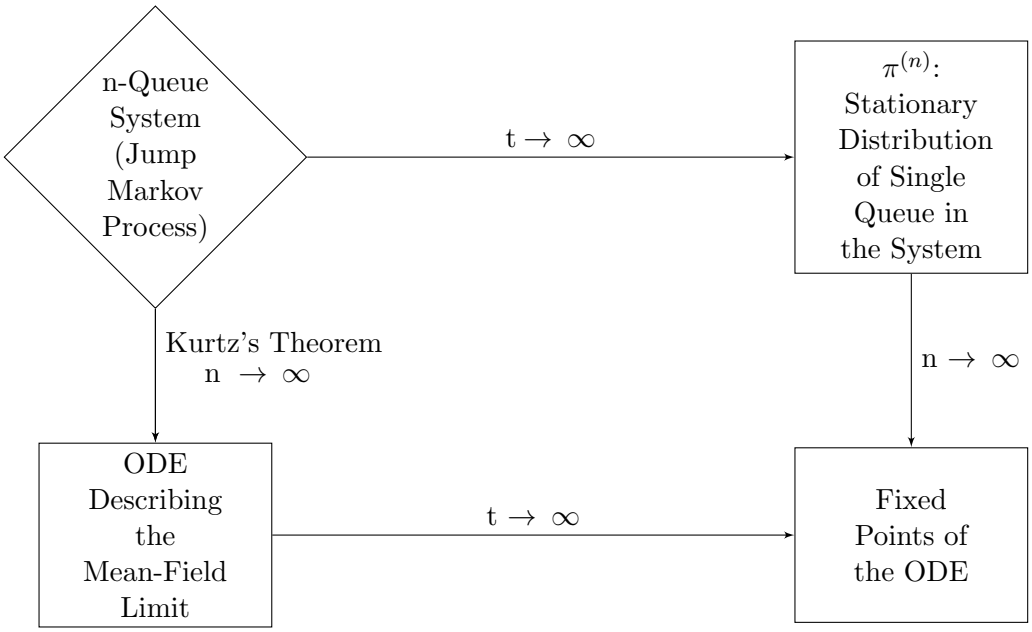


Figure 1: The Supermarket Model with $d = 2$

In order to analyze this model, we represent the state of the process as a jump Markov process. The jump Markov process corresponding to each $n-$server system can be shown to have a unique stationary distribution $\pi^{(n)}$, which represents the equilibrium empirical queue length distribution. However, unlike in the case of an M/M/1 queue, the distribution $\pi^{(n)}$ does not admit an explicit analytical expression. Instead, to gain insight into $\pi^{(n)}$ for

large n, we study the weak limit π of $\pi^{(n)}$, as n tends to infinity. This limit can be shown to exist and to admit a characterization as the unique stable fixed point of a countable system of ordinary differential equations (ODEs). We use Kurtz's theorem to show that the latter system of ODEs describes the finite-time evolution of the limit of the (suitably scaled) sequence of jump Markov processes that model the $n-$server system. Figure 2 illustrates the basic framework for our asymptotic analysis.

Figure 2: **Methods of Analysis**



1.1 The Classic Supermarket Model

The original "supermarket model" was completely analyzed by Vvedenskaya, Dobrushin, and Karpelevich [1] for the case $d = 2$, and the application of Kurtz's theorem and the analysis of the system of ordinary differential equations (ODEs) and their fixed points (see Figure 2) for arbitrary $d \geq 2$ was carried out by Mitzenmacher [2]. For any integer $d \geq 2$, the continuous version of the dynamics of the supermarket model can be described by the following system of ODEs: for all $t \geq 0$,

$$\begin{cases} \dfrac{ds_i}{dt}(t) = \lambda(s_{i-1}^d(t) - s_i^d(t)) - (s_i(t) - s_{i+1}(t)) & \text{for } i \geq 1, \\ s_0(t) = 1. \end{cases} \tag{1}$$

Here, s_i represents the limiting fraction of queues that have no less than i customers. The quantity s_i can decrease only when there is a departure from a queue with i customers.

Since the fraction of queues with exactly i customers is $(s_i - s_{i+1})$ and the service rate is 1, the rate of decrease of s_i is equal to $(s_i - s_{i+1})$. On the other hand, the quantity s_i increases only when there is an arrival into a queue with $i-1$ customers. Observing that the normalized arrival rate of customers is λ and that $(s_{i-1}^d - s_i^d)$ represents the probability that the shortest of d randomly sampled queues (with replacement) has $i-1$ packets, we then obtain the ODE in equation (1). The boundary condition $s_0(t) = 1$ trivially follows from the fact that at all times, every queue in the system has 0 or more packets.

It was also shown in [1] and [2] that the system of ODEs has a unique fixed point $\{s_i\}_{i=0}^{\infty}$, which is obtained by setting $ds_i/dt = 0$ in (1). Indeed, this fixed point is easily seen to take the following explicit form:

$$s_i \;=\; \lambda^{\frac{d^i-1}{d-1}}, \qquad \text{for } i \geq 0.$$

This expression says that for $d \geq 2$, the tail of the limiting stationary queue length distribution decreases double exponentially, which is an exponential improvement over the $d = 1$ case, in which the tail decreases geometrically, with $s_i = \lambda^i$.

Mitzenmacher goes on to provide simulations to test the accuracy of the model. For n=100, he focuses on the expected time a customers spends in the system by taking the average of 10 runs, each with 100,000 time steps (and ignoring the first 10,000 time steps), for d =2, 3, and 5. The following table shows how these simulation results compare to the results predicted by the ODE approximation. Note that the predicted approximation of the expected sojourn time of a customer, namely, the expected time a customer spends in the system (including in the queue and in service) is $\sum_{i=1}^{\infty} s_i^d \;=\; \sum_{i=1}^{\infty} \lambda^{\frac{d^i-d}{d-1}}$.

Table 1: **Average Time in the Supermarket Model**

d	λ	Simulation	Prediction	Rel. Error(%)
2	0.50	1.2673	1.2657	0.1289
	0.70	1.6202	1.6145	0.3571
	0.80	1.9585	1.9475	0.5742
	0.90	2.6454	2.6141	1.1981
	0.95	3.4610	3.3830	2.3028
	0.99	5.9275	5.4320	9.1227
3	0.50	1.1277	1.1252	0.2146
	0.70	1.3634	1.3568	0.4858
	0.80	1.5940	1.5809	0.8314
	0.90	2.0614	2.0279	1.6533
	0.95	2.6137	2.5351	3.1002
	0.99	4.4080	3.8578	14.2607
5	0.50	1.0340	1.0312	0.2637
	0.70	1.1766	1.1681	0.7250
	0.80	1.3419	1.3289	0.9789
	0.90	1.6714	1.6329	2.3564
	0.95	2.0730	1.9888	4.2363
	0.99	3.4728	2.9017	19.6825

This demonstrates the accuracy of the ODE approximation described above for arrival rates that are not too close to capacity (i.e. up to 95% capacity).

In conclusion, the papers [1] and [2] make the powerful observation that increasing the number of choices from $d = 1$ to $d = 2$ brings an exponential improvement in the

stationary sojourn time, while increasing the number of choices to $d \geq 3$ adds only a geometric improvement on top of that. This result has driven countless subsequent studies aimed at optimizing the load balancing problem. In the next section, we discuss various modifications of the original supermarket model that have been studied in the literature.

1.2 Outline of the Thesis

This thesis begins with a comprehensive survey of previously analyzed variations of the classic supermarket model in Section 2. We then introduce our threshold supermarket model in Section 3.1 and explain our motivation for studying such a system. The main results of our analysis can be found in Section 3.2, followed by our examination of an application of our model. Our analysis of this application, in which we explore the tradeoffs between performance and costs from sampling, can be found in Section 3.3. We also present simulations in Section 3.4, which show the accuracy of our proposed model. We then provide detailed derivations of our model and proofs of our main results, which can be found in Section 4. Finally, we introduce and analyze the associated threshold supermarket game. The description of the game, along with our main results and their implications, can be found in Section 5. A summary of our findings and remaining open questions conclude this thesis and can be found in Section 6.

2 Literature Review

2.1 Modifications of the Classic Supermarket Model

2.1.1 Migration Costs

The original supermarket model assumes that any migration of jobs incurs no cost or penalties. In their paper "Load Balancing with Migration Penalties" [3], Farias, Moallemi, and Prabhakar introduce a new system in which there is a cost for migration. They propose that after a customer arrives at a queue, the queue can either decide to keep and process the task or push the task to another server, transforming this $K = 1$ job into $K \geq 1$ independent jobs. In this way, the system captures a more realistic system in which the act of pushing a task has a cost to the overall throughput.

Because transferring a task to an alternate queue can now become costly, Farias et al. propose two possible criteria for determining whether or not a task should be pushed. The first criteria states that a job is transferred only if the sampled queue satisfies an additive threshold condition (that it has at least K jobs less than the original queue). The second criteria states that the job is transferred only if the sampled queue satisfies a multiplicative threshold condition. These threshold conditions are defined as follows:

$$\begin{aligned} &\text{Additive Threshold:} && q_i(t-1) > q_{i^*}(t-1) + C, \\ &\text{Multiplicative Threshold:} && q_i(t-1) > \max(\alpha q_{i^*}(t-1),\, q_{i^*}(t-1) + K). \end{aligned}$$

where K is the number of jobs sent to another queue in migration, i^* represents the index of the shortest of the d queues sampled with replacement, $C \geq K$, and $\alpha > 1$.Their goal is to identify simple migration policies that can reduce backlogs, while providing the highest throughput.

This model could be of relevance in a system where there exist specialists and generalists. In this scenario, a specialist can accomplish a specific task very quickly. Unfortunately, if jobs pile up and the queue reaches a certain threshold length, it may be forced to push a job to a shorter queue (that of a generalist) to reduce backlog. However, because the generalist lacks the specialist's specific expertise, the generalist will take longer to process the task. This scenario illustrates when a cost may appear from migration within a system. In brief, this variant of the supermarket model captures the notion of a preferred allocation, which is of relevance for caching in multiprocessor systems wherein each processor has a preferred cache [4].

After analyzing the associated ordinary differential equations and fixed points and simulations, Farias et al. found that the additive threshold policy is stable if $\lambda < \frac{1}{K}$, but can be unstable for some $\lambda \in \left(\frac{1}{K}, 1\right)$, thus showing that this policy leads to a loss in throughput. On the other hand, they showed that the multiplicative threshold policy is always stable, provides no loss of throughput, and provides improvements in average waiting time. This implies that sharing can provide improvements over the non-sharing case, but we ask: is there an optimal threshold to decide when to share?

2.1.2 Asymmetric Models

Vöking presents another interesting extension in his paper "How Asymmetry Helps Load Balancing" [5]. He begins by examining the effects of asymmetry, non-uniform sampling, and dependent sampling in a related balls and bins model. In the classic balls and bins model analyzed by Azar, Broder, Karlin, and Upfal, [6], d bins are sampled uniformly and independently at random, and the ball is assigned to the least loaded of the d bins, with ties broken uniformly at random. Vöcking expands upon this by looking at sampling non-uniformly and possibly dependently from the queues. To perform the analysis, he defines three different types of bin selection:

> 1) The classic model which samples d locations **uniformly and independently** at random. However, instead of ties being broken randomly (as is the case in [6]), any ties that may occur are resolved by the *Always-Go-Left* policy, which places the ball in the leftmost group of those classified as the least-loaded. This assymmetry is found to be irrelevant in this uniform case.
> 2) A model in which the d locations are sampled **non-uniformly and independently** at random. The n bins are divided into d groups of equal size and numbered from 1 to d. For each ball, d of

the bins are sampled as follows: The *ith* location, for $1 \leq i \leq d$, is sampled uniformly and independently at random from the *ith* group. The ball is then placed in the least loaded of the d bins, where any ties are broken according to the *Always-Go-Left* policy. This asymmetry, unlike the uniform case, provides a dramatic improvement in the non-uniform case. It should be noted however that this assignment requires the state of $\frac{n}{d}$, rather than just d, queues.
3) A model where the d bins are sampled **non-uniformly and possibly dependently** at random, where any ties are broken according to the *Always-Go-Left* policy.

To analyze the sequential allocation scheme with these three different bin selection methods, Vöcking examined the upper and lower bounds on the maximum load of the systems. For the case where $d = 1$, the introduction of sampling from a non-uniform distribution causes the expected maximum load to worsen. However, for $d \geq 1$, the upper bound on the maximum load in the non-uniform case (Case 2) greatly improves upon the upper and lower bounds of the uniform case (Case 1).

Uniform and Independent		**Non-Uniform and Independent**
$\ln \ln n / \ln d \pm \Theta(1)$	$>$	$\ln \ln n / (d \cdot \ln \phi_d) + O(1)$

where $\phi_d = \lim_{k\to\infty} \sqrt[k]{F_d(k)}$ and $F_d(k) = 0$ for $k \leq 0$, $F_d(1) = 1$, and F_d is a function defined recursively as follows: $F_d(k) = \Sigma_{i=1}^{d} F_d(k-i)$, for $k \geq 2$. This result also shows that the effect of d on the maximum load in the non-uniform case is linear (whereas the effect is only logarithmic in the original uniform case). Additionally, he found that the non-uniform and dependent scheme (Case 3) leads to an almost matching lower bound, proving that adding dependence produces no additional significant improvements.

Vöcking then went on to analyze the case where more balls exist than bins and where an infinite number of ball additions and deletions occur. He came to the following generalized theorem: "Supposing that at most $h \cdot n$ balls exist at any point in time. Then the *Always-Go-Left* algorithm yields maximum load $\ln \ln n/(d \cdot \ln \phi_d) + \mathrm{O(h)}$," where a ball is assumed to exist if it has entered the system but has not yet been deleted at time t.

This leads us to ask: How does asymmetry affect other extensions of the load-balancing problem? Does it yield improvements in models with an adaptive d?

2.1.3 Memory

Working with Prabhakar and Shah, Mitzenmacher expanded and improved upon his original model in a paper titled "Load Balancing with Memory" [7], published in 2002. This memory model has the same set-up as the supermarket model, described in the introduc-

tion, but instead of sampling d new queues with each arrival, only $d-1$ queues are sampled and compared with the memory of the shortest queue after placement of the previous arrival. This can be generalized to storing $1 \leq m \leq d$ of the least-loaded queues from the last round of sampling. Such a system is denoted as a (d,m)-memory policy, where d represents the number of new queues sampled uniformly and independently at random and m represents the number of queues stored in memory. For simplicity, Mitzenmacher et al. chose to focus on the case where $m = 1$. In addition, in order to draw comparisons to the commonly analyzed case with $d = 2$ choices, focus remained on a system where each new arrival is routed to the shorter of **one** random queue and **one** queue from memory. This scheme is labeled the (1,1)-memory policy.

In a previous paper, Shah and Prabhakar [8] found that when analyzing the supermarket model with service rates μ_i, where $\Sigma_i \mu_i = n$, and with sampling d locations with replacement (but without memory), the backlog of the system increases infinitely. However, by implementing the (1,1)-memory policy, the queue lengths remain bounded. The use of memory provides the ability to differentiate slower servers from faster servers and helps achieve stability. Shah and Prabhakar also utilized large deviations theory to find a bound for the maximum load in the (d,1)-memory policy with high probability.

This paper analyzes the same load balancing memory problem (in both the discrete and continuous cases) with memory but finds **exact** bounds. This is obtained by applying an extension of Kurtz's theorem. The system is modeled as two Markov chains– one for memory and one for servers. Simulations are also provided, which prove the accuracy of the model's predictions but also emphasizes the importance of the O(1) constants to the system's actual behavior. Average time spent in the system is the parameter of interest.

Mitzenmacher et al. found that implementing a (1,1)-memory policy provides a dramatic improvement over sampling two random queues. In fact, the bound of the maximum load matches that found by Vöcking's asymmetric *Always-Go-Left* scheme, which is as follows:

$$\frac{\ln \ln n}{2 \ln \phi} + O(1), \text{ where } \phi = (1 + \sqrt{5})/2, \text{ the golden ratio.}$$

Additionally, Mitzenmacher et al. [7] examined the same system with memory but added the component of truncation at a threshold level L. They defined and analyzed two different versions of this: 1) S_L^b where any arrival at a queue of length at least L is dropped from the system and 2) S_L^w where any arrival at a queue of length at least L ignores memory and resorts back to choosing d random queues. Surprisingly, both of these schemes have the same limit and lead to identical results as the original (d,1)-memory system.

In practice, these results provide a good approximation for systems with large bursts of tasks from a source, making the assumption of some memory reasonable. A common application is sticky routing, in which an efficient route, once identified, is used repeatedly until congestion occurs. Can combining asymmetry and memory provide even further improvements?

2.1.4 Load Stealing

Thus far, we have only examined models which incorporate work sharing. However, a very interesting topic analyzes the effectiveness of load stealing. In such a system, instead of a queue passing a task along to another when it becomes "overloaded," as in work sharing, a queue with room for capacity seeks out a task from overloaded queues. In theory, this model of load stealing is more efficient, because when all processors are busy, no migration attempts are made, reducing the system's work.

Mitzenmacher [9] analyzed several different versions of load stealing models by allowing the number of processors to approach infinity, developing families of differential equations to model the systems, and finding the fixed points of the systems. These limiting models are made possible through Kurtz's theorem as described above. With each system, Mitzenmacher also performed simulations to prove the accuracy of the models' predictions. These predictions become more accurate (and analysis is more interesting) in cases where the arrival rates are high. Therefore, for many of the systems described below, Mitzenmacher focused on arrival rates $\lambda = 0.90$ and $\lambda = 0.95$. This is interesting because it is in contrast to the "Power of Two Choices" [2], in which the predictions become worse as arrival rates increase.

He begins by analyzing a simple load stealing system, which has all the properties of the classic supermarket model with $d = 1$. Only now when a queue becomes empty, it independently and uniformly at random samples from the other queues and "steals" a job if the queue sampled contains more than one job. This simple stealing system proves stable and the tails decrease geometrically faster than the system without stealing. He slowly added more components to the model to make it more interesting and more applicable to real-life settings.

In the next system examined, a steal only occurs if the queue sampled has some minimum threshold T tasks (previously $T = 2$), allowing stealing to be more efficient. Again, the system proves to be stable with the tails decreasing at a geometrically faster rate. Adding on this threshold stealing, Mitzenmacher allowed for repeated steal attempts. That is: if the first queue sampled did not contain the minimum threshold of tasks, another queue would be sampled, and so on. These stealing attempts occur at rate r per unit of time. The time between attempts is assumed to be exponentially distributed. The next system modeled pre-emptive stealing, or the act of stealing a job when one's own queue had at most $B > 1$ tasks left. Intuitionally, this provides for a more even queue distribution. It only makes sense to allow a steal if the queue sampled contains more than B tasks. Therefore, Mitzenmacher chose the threshold $T \geq B + 2$.

He then went on to more complex (and more realistic) extensions of the model. These extensions increased the state space, making analysis much more difficult. The first interesting case explored involves constant service times, as opposed to the classical but unrealistic exponentially distributed service times. Mitzenmacher chose the system with a sequence of c stages of constant service. The random variable of the constant service time

of each stage is independent and exponentially distributed with mean $\mu = \frac{1}{c}$. In this way, as $c \to \infty$, the average time in these c stages is 1. Mitzenmacher chose the case where $c = 20$ and found that the constant service times system performs significantly better than that of the exponential service time. Another interesting, yet complicated system is adding a transfer delay. Instead of a transfer occurring instantaneously, as assumed before, transfers occur at rate r. The transfer delay is exponentially distributed with mean $\mu = \frac{1}{r}$. He applies this transfer delay to the system where stealing occurs only when a queue is empty and the queue sampled has a minimum T tasks. A queue can only steal one task at a time. In other words, the queue cannot steal again when a task is already on the way. Because arrivals may occur at a queue while a steal is being transferred to it, the threshold T must be chosen carefully to ensure stealing is efficient. Therefore, to minimize the expected time for a task, the best choice is $T = \frac{1}{r} + \frac{\lambda}{r}$. Transfer delays, even if they are very short, are found to have a very large negative impact on performance. The significant effects are quite surprising.

Mitzenmacher then explored the possibility of choice in stealing, where d queues are sampled before a steal occurs from the most loaded queue, assuming at least one meets the minimum threshold T. Although adding choice does improve the system, the majority of the gain from stealing comes from $d = 1$. Mitzenmacher concluded that in real systems, adding choice to stealing is not worthwhile. Lastly, he analyzed the case where multiple steals can occur. Intuitionally, this can be appropriate and reduce average time spent in the system if the threshold T is high. Mitzenmacher considers the case where $k \leq \frac{T}{2}$ tasks are stolen at a time. This and other similar variations of k equalize the loads and do in fact improve performance.

In conclusion, Mitzenmacher found that as predicted, load stealing models improve performance (compared to those without). This is evidenced by the fact that the expected time spent in the system decreases and the tails decay at a geometrically faster rate. Having $d \geq 2$ choices when stealing also provides improvement; however, the majority of the gains is already seen when $d = 1$. Additionally, adding a transfer delay with rate r greatly hurts the system's performance, even when delays are small. Mitzenmacher also proposes other interesting load stealing systems to analyze. The first is allowing the processors to have heterogeneous service times. He describes a case where fast and slow processors can be distinguished between when making choices. Another interesting system he describes includes differentiating between internal and external arrival rates, where internal arrivals refer to jobs "spawned" by tasks already at a processor. He also expresses his interest in developing a general framework that will help with the complex convergence issues he faced in many of these more complicated load stealing models.

2.1.5 Pull vs. Push

Minnebo and Van Houdt recently published a paper titled "A Fair Comparison of Pull and Push Strategies in Large Distributed Networks" [10], in which they further explore the

advantages and disadvantages of pull (load stealing) and push (load sharing) strategies. By allowing the number of servers to approach infinity, they compared the average time spent in the system under pure push strategies, pure pull strategies, and hybrid strategies. The system analyzed is slightly different than the one examined by Mitzenmacher. Instead of attempting a push/pull after the completion or arrival of a task, an idle queue in a pull system makes attempts at some rate r and a queue with jobs waiting in a push system make attempts at some rate r. They introduce a new and unique tool of analysis: normalizing the different strategies by setting the overall probe rate R of each equal. Here R is defined as the average number of push/pull attempts sent by a queue per unit of time. Given the maximum overall probe rate R and arrival rate λ, Minnebo and Van Houdt found that push strategies actually perform better than pull strategies for $R > 0$ and $\lambda < \phi - 1$, where $\phi = \frac{1+\sqrt{5}}{2}$, the golden ratio. The push strategy performs better than the pull strategy if and only if $2\lambda < \sqrt{(R+1)^2 + 4(R+1)} - (R+1)$. Additionally, hybrid strategies always perform worse than either a pure pull or pure push strategy. This analysis, however, only examines the case where push attempts begin when there is at least one task in the queue, and pull attempts begin only when a queue becomes idle. Additionally, a queue only accepts a job if it is currently idle. There are many natural extensions of this still to be explored, such as attempting to push only when some fixed minimum threshold $T > 1$ is reached, or preemptively attempting to pull, or accepting a job at some threshold $T > 0$. Minnebo and Van Houdt presented some interesting new parameters to focus on when comparing push and pull systems, but left much room for further research.

2.1.6 Centralization

Another interesting topic to consider is centralization. In a recent paper titled "On the Power of (even a little) Centralization in Distributed Processing" [11], Tsitsiklis and Xu examine the effectiveness of centralization. The focus remains on systems with many local stations, which can only serve tasks addressed specifically to them. They add the idea of sending a fraction p of resources to serve in a centralized manner, such as serving the most-loaded station. They describe two separate applications of interest.

The first is a group of local stations which are connected to a centralized processor that can serve the station with the largest queue whenever possible. A key question is: what is the optimal fraction p of resources to devote to the central server? Because local servers cannot transfer tasks between each other, it logically makes sense to give more resources to a central server. Doing this would prevent one station from being empty while another has a large and growing queue. However, this would require much more communication. Do the performance improvements outweigh the overhead costs? They also wonder if adding a little centralization is any different than none.

The second application is a group of local stations, which are all serviced by one central server. Optimally, the server pulls from the station with the greatest load. However, these communications can be very costly and in some cases, not possible. In such scenarios, a

task is pulled uniformly at random from the local stations. If the station sampled is empty, it is a waste of that turn. What is the optimal level of communication to make this process efficient?

Both of the described applications can be modeled by the same mathematical structure and are addressed together in the paper. The major result, found through the mean field approach, states that adding (any) centralization ($p > 0$) allows the average queue length to scale as $log_{\frac{1}{1-p}} \frac{1}{1-\lambda}$, which is exponentially smaller than the classic case with scaling of $\frac{1}{1-\lambda}$. A small amount of centralization has a large impact on performance.

An interesting extension of this paper would be to look at centralization with transfer delays/costs. This would be much more realistic as the applications of the local stations most often occur with stations set apart by geography. If this were the case, the time costs of transferring tasks would presumably have a large impact. Additionally, this paper also leaves much room for examining centralization in systems with general service distributions and disciplines.

2.1.7 Cost-Aware Monitoring

The majority of the models discussed do not consider a significant cost from monitoring and information gathering in the migration of jobs. Breitgand, Cohen, Nahir, and Raz [12] develop several self-adaptive algorithms, for both centralized and fully distributed systems, which model the trade-off between the gains from monitoring and the costs of obtaining information about the system. They define this new system, aimed at maximizing utility for management, as the Extended Supermarket Model (ESM). They have demonstrated the accuracy of their model and predictions via both simulations and a real testbed. Through analyzing the system, they have found the optimal number of servers to monitor in order to achieve the minimum average time spent in the system with the optimal cost at each service rate. Their cost-aware model has proven to provide dramatic improvements over the cost-oblivious counter-part.

They describe a system, which has costs when a load request is received either from other servers (when completing the initial d sampling upon an arrival) or a monitor. Load requests are immediately responded to, therefore taking time away from actual service and affecting service time. This time includes what it takes to receive the request, parse it, retrieve the necessary information, and respond. Let m represent the mean response time to a monitoring request. Then $C = \frac{\mu}{n} = \frac{1}{n}$ represents the impact on service time. Logically, this cannot be viewed as negligible and should be considered in load balancing problems. Because the mean service time is normalized to 1, the effective service rate must be smaller. Because each new arrival creates d load queries, the effective service rate is:

$$\mu' = 1 - \lambda \cdot d \cdot C.$$

Subsequently, let $\rho = \frac{\lambda}{\mu'}$ be the arrival rate adjusted for the effective service rate. Therefore, it must be the case that $\lambda < \frac{1}{1+d \cdot C}$. This shows right away that when the load is high,

monitoring can be unwise and cause an unstable system. However, when the load is low enough, monitoring can be a very powerful tool.

Breitgand at el. present two different forms of a centralized system. The first utilizes periodic updates, which are stored in a centralized database. When new arrivals occur, the information on the d queues sampled is looked up on this central database. Q updates occur per unit of time. Checking more often provides greater accuracy of information in the database but increases communication costs greatly. The second form is per-job polling. When examining the centralized case, Breitgand et al. found that contrary to popular thought, there is no significant difference between periodic updates and per-task polling when the optimal d is chosen. Although it is true that when d is large, per-job polling greatly outperforms periodic updates. In the adaptive algorithm, a centralized router dynamically estimates the load $(\lambda \cdot n)$ and the overhead efficiency ratio C. With this information, an optimal d is pulled from a look-up table. Subsequently, the task is sent to the least-loaded of the d queues sampled.

For the fully-distributed case, two different main algorithms were developed. The first relied on a pre-determined look-up table, as in the centralized case. This look-up table indicates the optimal d based on dynamic estimates of the current load $(\lambda \cdot n)$ and the current overhead efficiency ratio C. The alternate algorithm consists of each server dynamically performing a cost-benefit analysis of forwarding the job received. If the cost of forwarding is determined to be greater than the expected gains, the task will be routed to the least-loaded of the queues previously sampled. Forwarding a task now costs the time it takes to transfer the task (CT) and the CPU cost of analyzing the effectiveness of forwarding. These algorithms provide a great framework for explicitly analyzing systems with non-negligible communication and decision-making costs.

The opportunities to expand upon this research and apply other forms of costs to the load balancing problem are numerous and fascinating. The Extended Supermarket Model provides an effective way of quantifying costs of monitoring and greatly enhances ability to analyze systems with non-negligible costs. This concept of explicitly evaluating cost in models is intriguing and very applicable to similar game theory related applications of the supermarket model.

2.2 Other Generalizations

The analysis of such randomized load balancing algorithms is still a very active area of current research. Recent works have considered other disciplines besides FIFO, such as last-in first-out preemptive resume (LIFO-PR) and process-sharing (PS) [13] and [14], as well as more general service distributions [15], and many open problems remain.

3 The Threshold Supermarket Model

3.1 Description of the Model

Recall that the supermarket model describes a system in which customers arrive to a service center with n parallel queues according to a Poisson process with rate λn, where $\lambda < 1$. Each queue has a dedicated server that processes tasks according to a FIFO service discipline. Service times are i.i.d. exponentially distributed with mean $\mu = 1$. After sampling d of the n queues independently and uniformly at random (with replacement) upon arrival, each customer joins the shortest queue, with any ties being broken uniformly at random.

In this thesis, we analyze a variant of the supermarket model in which the number of queues sampled is determined according to the following threshold function:

$$\#\text{ of queues sampled } = \begin{cases} 1 & \text{if } q < T, \\ d & \text{if } q \geq T, \end{cases} \tag{2}$$

where q is defined to be the observed length of the initial queue sampled. This thesis focuses on the case where $d = 2$.

Note that when $T = 0$, our model reduces to the supermarket model of [1] or Mitzenmacher's [2] "The Power of Two Choices," and the number of queues sampled becomes a fixed constant d. In such a case, it is well known that increasing the number of queues polled from $d = 1$ to just $d = 2$ improves waiting time exponentially and captures the majority of the gain from allowing $d \geq 2$. However, sampling $d = 2$ queues with every arrival can be very costly to the system. So a logical question to ask is: Does the system require $d = 2$ queues to be sampled for every customer, or are there some circumstances in which $d = 1$ is sufficient? If the length of the first queue sampled is "small enough," can the extra sampling be eliminated without either hurting the total throughput of the system or leading to a significant deterioration in the expected steady state sojourn time? If so, what threshold is appropriate? These questions led to the development of our threshold model. In this thesis, we seek to answer the question: Given a marginal cost for sampling, does there exist an optimal threshold $T^* > 0$ which minimizes the overall cost of the algorithm, measured as the sum of the expected waiting time and total cost of sampling?

The proposed system is designed as follows. Customers arrive according to Poisson process with rate λn, where $\lambda < 1$. Upon arrival, each customer samples one queue uniformly at random. If the queue length exceeds or is equal to some threshold T, the customer samples an additional queue and joins the shorter of the $d = 2$ queues sampled. However, if the initial queue length observed is below the threshold T, the customer simply joins this queue, eliminating the cost that would have resulted if an additional queue had been sampled. The logic behind such a system is as follows: when a queue is short enough (less than T), the benefit of sampling an additional queue may be outweighed by the cost of extra polling. Therefore, by finding the optimal threshold T^*, the system will effectively

be eliminating unnecessary search costs and optimizing the efficiency of the system.

We now introduce some common notation that will be used throughout our analysis. So far in our analysis, we have already assigned n to be the number of servers in our system, d to be the number of queues sampled by each customer, and T to be the threshold value that determines d. Additionally, we define $m_i^{(n)}(t)$ to be the number of queues with at least i customers at time t, and $s_i^{(n)}(t) = \frac{m_i^{(n)}(t)}{n}$ to be the fraction of queues with at least i customers at time t. For convenience, the $s_i^{(n)}(t)$ notation will be used, and the reference to time t will be excluded when the meaning is clear.

3.2 Main Results

Define $\overline{\mathcal{S}}$ to be the space of sequences $x = \{x_i\}_{i=0}^{\infty} \in [0,1]^{\infty}$, such that:

$$x_0 = 1 \geq x_1 \geq \dots \; x_k \geq \dots$$

and equip $\overline{\mathcal{S}}$ with the metric:

$$\rho(x, x') = \sup_{i>0} \frac{|x_i - x_i'|}{i}, \quad x, x' \in \overline{\mathcal{S}}.$$

Let $\mathcal{S}$ be the subspace of the metric space $\overline{\mathcal{S}}$ defined by:

$$\mathcal{S} = \{s \in \overline{\mathcal{S}} : \sum_{i=1}^{\infty} s_i < \infty\}.$$

Also, let $\mathcal{S}^n = \mathcal{S} \cap \frac{1}{n}\mathbb{Z}^n$. When the number of servers is n, we shall encode the state of the system in the threshold model in the countable vector-valued process $s^{(n)}(t) = (s_0^{(n)}(t),\, s_1^{(n)}(t),\, \dots,\, s_k^{(n)}(t),\, \dots)$. Note that for each $n \in \mathbb{N}$ and $t \geq 0$, $s^{(n)}(t)$ lies in $\mathcal{S}^n$.

Theorem 1. *For each $n \in \mathbb{N}$, $\{s^{(n)}(t),\, t \geq 0\}$ is a jump Markov process that is ergodic and has a unique stationary distribution $\pi^{(n)}$.*

Our goal is to find an approximation for this unique stationary distribution $\pi^{(n)}$. To do so, we first show that for large n, the evolution of $s^{(n)}$ can be approximated by the unique solution s of the following coupled system of ODEs: for $t \geq 0$,

$$\begin{cases} \frac{ds_i}{dt}(t) = \begin{cases} \lambda(s_{i-1}(t) - s_i(t))(1 + s_{\mathrm{T}}(t)) - (s_i(t) - s_{i+1}(t)) & \text{if } T > i-1, \\ \lambda(s_{i-1}^2(t) - s_i^2(t)) - (s_i(t) - s_{i+1}(t)) & \text{if } T \leq i-1, \end{cases} \\ s_0 = 1, \\ s_1 = \lambda, \end{cases} \tag{3}$$

with appropriate initial condition $s(0) \in \mathcal{S}$.

Note that when $T = 0$, this reduces to the system of ODEs (1) associated with the supermarket model.

(See Section 4.1 for full derivation.)

Upon analyzing this system, we come to our next result.

Theorem 2. *Given any $s_0 \in \overline{\mathcal{S}}$, the system of ODEs has a unique solution $s(t) = \{s_i(t)_{i=0}^{\infty}, t \geq 0\}$, with initial condition $s(0)$. Moreover, if $s(0) \in \mathcal{S}$, then $s(t) \in \mathcal{S}$ for all $t \geq 0$. For any sequence $s^{(n)}(0) \in \mathcal{S}^n$, $n \in \mathbb{N}$, such that*

$$\lim_{n\to\infty} \rho(s^{(n)}(0), s(0)) = 0,$$

for every $R < \infty$, we have

$$\lim_{n\to\infty} \sup_{t\in[0,R]} \rho(s^{(n)}(t), s(t)) = 0.$$

(See Section 4.1 for detailed proof.)

We now study fixed points of the system of ODEs for the threshold supermarket model, which are obtained by setting $ds_i/dt = 0$ in (3). These lead to algebraic equations which, unlike in the case of the original supermarket model, lead to an implicit recursion due to the presence of s_T in the equations for s_i for $i < T$. The full details of this derivation are given in Section 4.2. In a nutshell, each term s_i in the solution of the recursion that describes the fixed point can be expressed in terms of (the a priori unknown) s_T, which itself is then characterized as the stable root of a certain function H, which we now describe. The function H is of the form $H = H_1 - H_2$, where

$$\begin{aligned} H_1(s_{\mathrm{T}}) &= (1 + s_{\mathrm{T}})^T, \\ H_2(s_{\mathrm{T}}) &= \frac{s_{\mathrm{T}} - \lambda s_{\mathrm{T}}^2}{\lambda^T(1-\lambda)}. \end{aligned} \tag{4}$$

Definition 1. *We will say s_{T}^* is a root of H if $H(s_{\mathrm{T}}^*) = 0$. Moreover, a root s_{T}^* is said to be stable if $H'(s_{\mathrm{T}}^*) < 0$.*

Lemma 1. *If $\lambda < \frac{1}{2}$, H has a unique root in $[0, 1]$. If $\lambda \geq \frac{1}{2}$, H has two roots in $[0, 1]$, of which only one is stable.*

(See Section 4.2 for proof.)

Remark 1. *By Lemma 1, we know that H has a unique stable root, which we will denote by s_{T}^*.*

Table 2 summarizes the numerically computed values of s_{T}^* for different values of λ and T.

Table 2: s_T^*

λ	Threshold	s_T^*
0.40	2	0.1290
	3	0.0446
	4	0.0165
	5	0.0064
0.50	2	0.2000
	3	0.0828
	4	0.0368
	5	0.0172
0.70	2	0.4050
	3	0.2232
	4	0.1282
	5	0.0774
0.80	2	0.5517
	3	0.3505
	4	0.2243
	5	0.1490
0.90	2	0.7431
	3	0.5618
	4	0.4087
	5	0.2994
0.95	2	0.8616
	3	0.7323
	4	0.5887
	5	0.4631

We are now ready to state the main result for the threshold model.

Theorem 3. *As $n \to \infty$, the stationary distribution $\pi^{(n)} = (\pi_i^{(n)})_{i=0}^{\infty}$ of $s^{(n)}$ converges to $\pi \in \mathcal{S}$, where $\pi_0 = 0$, $\pi_1 = \lambda$, and*

$$\pi_i = \begin{cases} \left[\dfrac{1 - \lambda}{1 - \lambda(1+s_T^*)}\right] \lambda^i (1+s_T^*)^i + \left[\dfrac{\lambda - \lambda(1+s_T^*)}{1 - \lambda(1+s_T^*)}\right] & \text{for } 2 \le i \le T+1, \\ s_T^* (\lambda s_T^*)^{2^{i-T}-1} & \text{for } i > T+1, \end{cases} \tag{5}$$

where s_T^* is the unique stable fixed point of H.

(See Section 4.2 for detailed proof and derivation.)

3.3 Finding the Socially Optimal Threshold: Algorithm & Numerics

We now examine an application of the main result: analyzing the tradeoffs in performance associated with eliminating the extra cost of unnecessary polling. We seek to identify the optimal threshold that minimizes the cost per customer, where given a threshold T, arrival rate λ, and a unit cost of sampling c_s, the cost is defined to be the sum of the limiting

average size of the queue length and the total sampling cost. Thus, the expected total cost of each customer can be represented by the following equation:

$$\begin{aligned} Cost(T) &= \mathbb{E}[\text{queue length}] \ + c_s \mathbb{E}[\text{number of extra searches}], \\ &= \sum_{i=1}^{\infty} s_i \ + \ c_s s_{\mathrm{T}}. \end{aligned} \tag{6}$$

The optimal threshold T^* can be found by minimizing this cost function with respect to T (that is, $T^* = argmin_{\mathrm{T}} Cost(T)$). Given a λ and the cost of sampling, numerically solving for T^* is trivial.

The following thresholds were found to be socially optimal given various values for λ and sampling cost c_s:

Table 3: **Optimal Threshold**

λ	c_s	T^*
0.40	0.1	1
	0.5	2
	1.0	3
	1.5	4
	2.0	5
0.50	0.1	1
	0.5	2
	1.0	3
	1.5	3
	2.0	4
0.70	0.1	1
	0.5	2
	1.0	3
	1.5	3
	2.0	4
0.80	0.1	1
	0.5	2
	1.0	3
	1.5	3
	2.0	4
0.90	0.1	1
	0.5	2
	1.0	3
	1.5	3
	2.0	4
0.95	0.1	1
	0.5	2
	1.0	3
	1.5	3
	2.0	4

These results indicate that for a sampling cost $c_s > 0$, it is always more efficient to implement a threshold $T > 0$. In other words, the performance of our threshold model always performs better than the classic "Power of Two Choices" [2] case where $T = 0$. Additionally, these results show that as the marginal cost of sampling increases, the benefit of implementing a threshold model increases with it and becomes quite large. The following figures compare the average cost per customer in the case without a threshold (when $T = 0$) to our case where an optimal threshold T^* is used to determine the number of queues sampled. As can be seen, for all λ and cost structures, our threshold model results in a lower average cost per customer and therefore, a more efficient system.

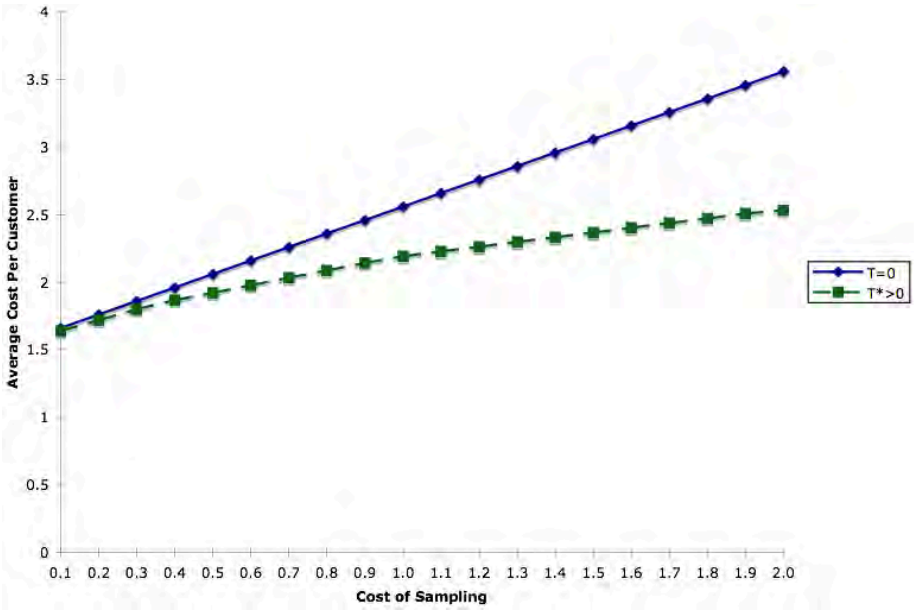
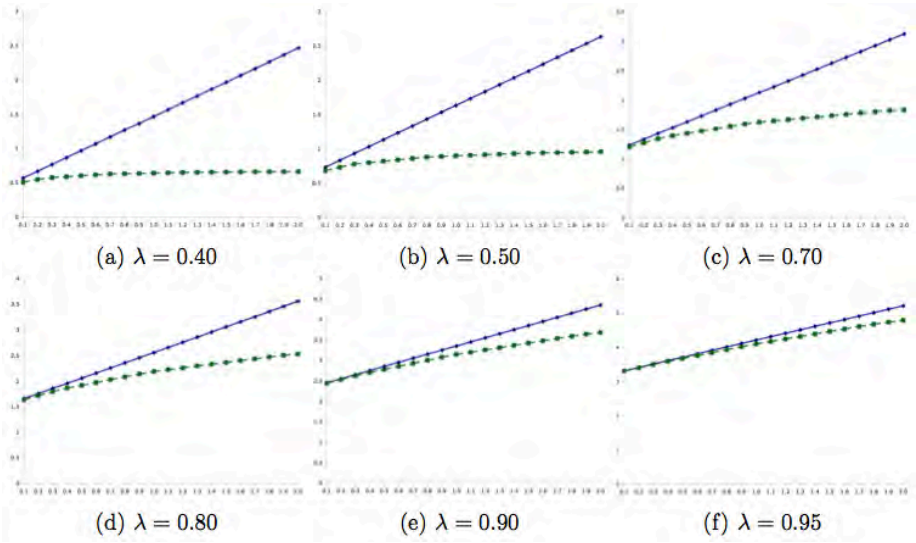


Figure 3: Detailed Comparison for $\lambda = 0.80$



(a) $\lambda = 0.40$ (b) $\lambda = 0.50$ (c) $\lambda = 0.70$

(d) $\lambda = 0.80$ (e) $\lambda = 0.90$ (f) $\lambda = 0.95$

Figure 4: General Comparison

The following figures show how the gains from using the optimal threshold T^* scale as we approach heavy traffic. As can be seen, as arrival rates increase, the gains from using the optimal threshold T^* decrease (but do not disappear). This intuitively makes sense. For a fixed threshold T^*, if one increases λ, then the probability of finding a queue length above T^* increases, and thus the model behaves more like the supermarket model. This suggests that it may be appropriate to scale T^* in an appropriate way with λ.

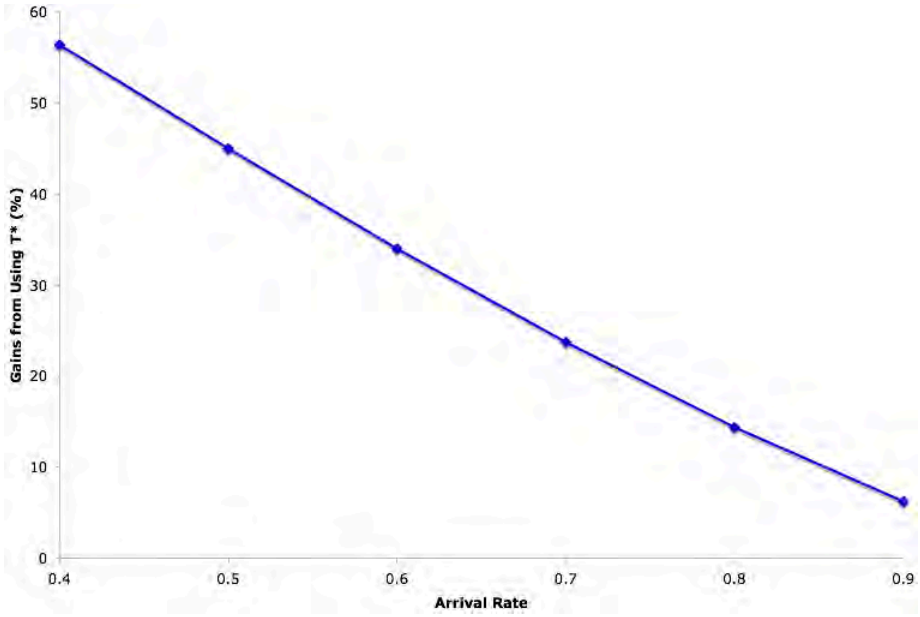


Figure 5: Percentage Gain for $c_s = 1$

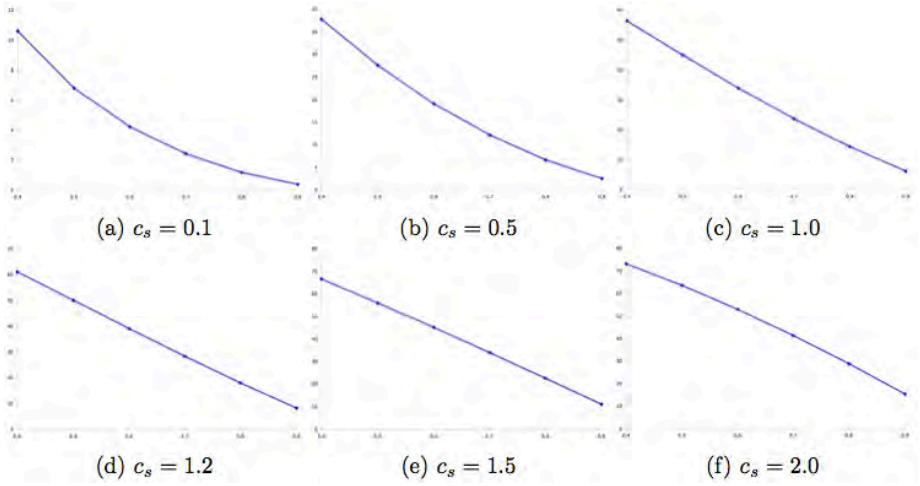


Figure 6: General Percentage Gain

3.4 Simulations

We simultaneously ran simulations of our threshold supermarket model with $n = 100$. Again, we focused on the expected total cost of each customer as a proxy for system performance. The following results are achieved by taking the average of ten trials, where each trial looks at time steps 10,000 to 100,000. The first 9,999 steps are not considered in order to let the system reach equilibrium. The following table shows how remarkably close the simulations are to predictions.

Table 4: **Simulations: Total Cost Comparisons**

λ	c_s	T^*	$Simulation\ C(T^*)$	$Prediction\ C(T^*)$	$Rel.Error(\%)$
0.40	0.1	1	0.5085	0.5056	0.5736
	0.5	2	0.6013	0.6002	0.1833
	1.0	3	0.6380	0.6392	0.1877
	1.5	3	0.6495	0.6543	0.7336
	2.0	5	0.6528	0.6609	1.2256
0.50	0.1	1	0.6857	0.6828	0.4247
	0.5	2	0.8201	0.8202	0.0122
	1.0	3	0.9002	0.8983	0.2115
	1.5	3	0.9344	0.9396	0.5534
	2.0	4	0.9504	0.9583	0.8244
0.70	0.1	1	1.2046	1.2001	0.3750
	0.5	2	1.4388	1.4315	0.5100
	1.0	3	1.6277	1.6252	0.1538
	1.5	3	1.7496	1.7368	0.7370
	2.0	4	1.8355	1.8352	0.0163
0.80	0.1	1	1.6438	1.6379	0.3602
	0.5	2	1.9309	1.9203	0.5520
	1.0	3	2.2026	2.1909	0.5340
	1.5	3	2.3979	2.3662	1.3397
	2.0	4	2.5264	2.5309	0.1778
0.90	0.1	1	2.4245	2.4427	0.7451
	0.5	2	2.7620	2.7803	0.6582
	1.0	3	3.1276	3.1446	0.5406
	1.5	3	3.4208	3.4256	0.1401
	2.0	4	3.6505	3.6828	0.8771
0.95	0.1	1	3.2305	3.3089	2.3694
	0.5	2	3.5982	3.6765	2.1297
	1.0	3	4.0341	4.0993	1.5905
	1.5	3	4.4201	4.4654	1.0145
	2.0	4	4.7527	4.7938	0.8574

We also performed the same analysis (comparing the results from $T = T^*$ to $T = 0$) as was done using the numerics in Section 3.3. Figures 7 and 8 depict the results.

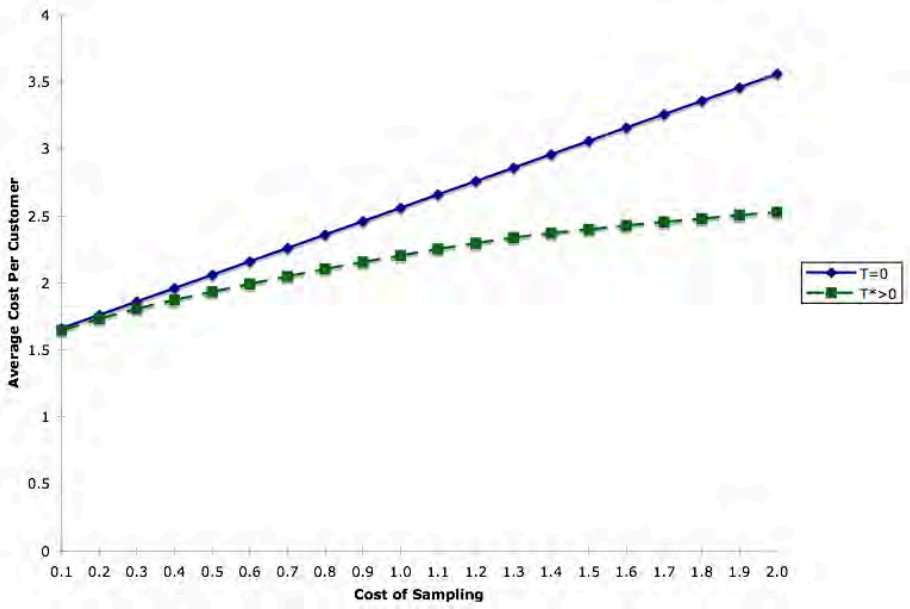


Figure 7: Simulations Comparison for $\lambda = 0.80$

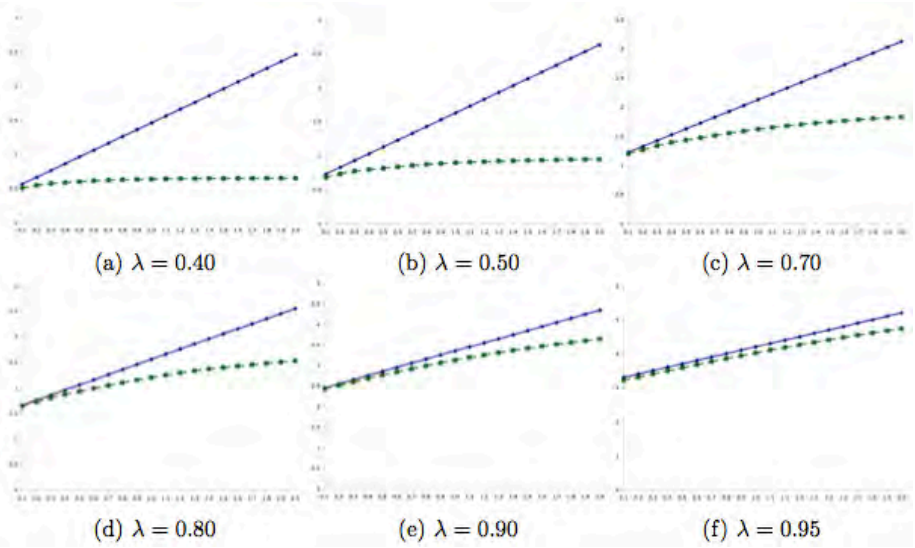


Figure 8: General Simulations Comparison

As can be seen, these results mirror those found from the numerics. That is, for any cost greater than zero, our threshold model always outperforms the parallel model without a threshold. Again, as the cost of sampling increases, the benefits from using a threshold increase as well.

4 Proofs of Theorems

4.1 Convergence to the System of Ordinary Differential Equations

Recall the state process $s^{(n)}$ lies in $\mathcal{S}^n$ and was defined in Section 3.2. To repeat: $s_i^{(n)}(t)$ is the random variable representing the fraction of queues in the n-server system that have i or more packets at time t. Given our assumptions of Poisson arrivals and exponential service times, it follows that $s^{(n)}$ is a jump Markov process with jump directions lying in $\mathcal{V}/n$, where $\mathcal{V} = \{\pm e_i,\ i \geq 1,\ \ldots,\ k,\ \ldots\}$ and e_i is defined to be the unit vector in $[0,\ 1]^\infty$ that has a 1 at the ith coordinate and 0 otherwise. For $s \in \mathcal{S}^n$, the transition $s \to s + \frac{e_i}{n}$ represents the event that an arriving job joins a queue with $i-1$ jobs.

Suppose $1 \leq i - 1 < T$. Then, this transition can happen in two ways. The first way is if there is a new arrival and the first queue sampled has $i-1$ jobs. Since $i - 1 < T$, this new arrival will then join this queue. When the system is in state $s \in \mathcal{S}^n$, the fraction of queues with exactly $i-1$ jobs is $(s_{i-1} - s_i)$, and so this event occurs at rate $n\lambda(s_{i-1} - s_i)$. The second way the transition $s_i \to s_i + \frac{e_i}{n}$ can happen is if there is a new arrival and the first queue sampled has T or more jobs (which happens with probability s_T). In this case, the policy dictates that a second queue should also be sampled, and the probability that the second queue sampled has $i-1$ jobs is again $(s_{i-1} - s_i)$. Moreover, since $i - 1 < T$, the arriving job, in this case, will join the queue with $i-1$ packets, thus increasing the number of queues with i packets.

Now consider the case when $i - 1 \geq T$. In this case, as in the original supermarket model, the transition $s \to s + \frac{e_i}{n}$ occurs if and only if an arrival samples two queues, and the smaller of the two queue lengths has $i-1$ jobs. This happens at rate $n\lambda(s_{i-1}^2 - s_i^2)$.

Finally, the transition $s \to s - \frac{e_i}{n}$ occurs due to departures from a queue with i jobs. Since the service rate is $\mu = 1$, this happens at a rate proportional to the number of queues with i jobs, which is $n\lambda(s_i - s_{i+1})$.

Putting this all together, we see that $s^{(n)}$ is a jump Markov process with the following jump directions and rates:

Jump Direction		**Jump Rate**
$+\dfrac{e_i}{n}$,	if $1 \leq i < T+1$,	$n\lambda(s_{i-1} - s_i)(1 + s_{\mathrm{T}})$,
$+\dfrac{e_i}{n}$,	if $T+1 \leq i$,	$n\lambda(s_{i-1}^2 - s_i^2)$,
$-\dfrac{e_i}{n}$,	if $1 \leq i$,	$n\lambda(s_i - s_{i+1})$.

The generator $\mathcal{L}^n$ of the Markov process $s^{(n)}$ thus takes the form:

$$\mathcal{L}^n f(s) = n \sum_{i=1}^{T} \left(\lambda(s_{i-1} - s_i)(1 + s_{\mathrm{T}}) \left[f(s + \frac{e_i}{n}) - f(s) \right] \; - \; \lambda(s_i - s_{i+1}) \left[f(s - \frac{e_i}{n}) - f(s) \right] \right)$$
$$+ n \sum_{i=T+1}^{\infty} \left(\lambda(s_{i-1}^2 - s_i^2) \left[f(s + \frac{e_i}{n}) - f(s) \right] \; - \; \lambda(s_i - s_{i+1}) \left[f(si - \frac{e_i}{n}) - f(s) \right] \right),$$

for all bounded functions $f : \mathcal{S}^n \to \mathbb{R}$ and $s \in \mathcal{S}^n$.

Proof of Theorem 1. Theorem 1 can be established using methods exactly analogous to those used by Vvedenskaya, Dobrushin, and Karpelevich in Theorem 5 of their paper [1]. We do not provide the details here.

In order to prove Theorem 2, we first state Kurtz's theorem.

Theorem 4. ***(Kurtz's Theorem):*** *Suppose that* $max_i |e_i| = \bar{e} \in (0, \infty)$ *and*

$$\max_i \sup_{x \in \mathcal{S}} r_i(x) \; < \; \infty.$$

Further, suppose we have a density dependent family satisfying the Lipschitz condition

$$|F(x) - F(y)| \; \leq \; K|x - y|,$$

for some constant K. Suppose $lim_{n \to \infty} X(0) = x_0$ *and let X be the deterministic process:*

$$X(t) \; = \; x_0 \; + \; \int_0^t F(X(u))du, \; t \geq 0.$$

Then,

$$\lim_{n \to \infty} \sup_{u \leq t} |X_n(u) \; - \; X(u)| \; = \; 0 \; a.s.$$

As stated, Kurtz's theorem does not directly apply here because we have a countable (and not a finite) number of jump directions. However, it turns out that we can extend the result to our problem using methods similar to those used in [1] and [2], and with the Euclidean metric replaced by the metric ρ introduced in Section 3.2. Below, we ignore the finiteness constraint, and verify the conditions of Kurtz's theorem.

Proof of Theorem 2. We must:

i) Prove that: $max_i |e_i| = \bar{e} \in (0, \infty)$.

Recall that the possible transitions are in $\mathcal{V}/n$, where $\mathcal{V} = \{\pm e_i : i \geq 1\}$ and e_i is defined to be the standard unit vector. It is clear that $max_i |e_i| = \frac{1}{n} \in (0, \infty)$.

ii) Prove that: $\max_i \sup_{x\in\mathcal{S}} r_i(x) < \infty$.

We set r_{+i} and r_{-i} to be the arrival rate and departure rate, respectively. Recall that these rates are defined in Section 4.1. Because $0 \le s_i \le 1$ for all i, $max\, r_{-i} = 1$. Additionally, $max\, r_{+1} = \lambda(1)[\mathbb{1}_{\{\mathrm{T}>i-1\}}(2) + \mathbb{1}_{\{\mathrm{T}\le i-1\}}(2)] = 2\lambda$. Therefore, $\max_i \sup_{x\in\mathcal{S}} r_i(x) = max\{1,\, 2\lambda\} < \infty$.

iii) Prove Lipschitz Condition holds: $|F(x) - F(y)| \le K|x - y|$ for some constant K.

$$F(x) = \sum_{i=1}^{\infty} \lambda(s_{i-1} - s_i)[\mathbb{1}_{\{\mathrm{T}>i-1\}}(1 + s_{\mathrm{T}}) + \mathbb{1}_{\{\mathrm{T}\le i-1\}}(s_{i-1} + s_i)] - (s_i - s_{i+1}).$$

Let $x = x_i$ and $y = y_i$ be two states of the model. Then,

$$\begin{aligned}|F(x) - F(y)| \le \sum_{i=1}^{\infty} |&\lambda(x_{i-1} - x_i)[\mathbb{1}_{\{\mathrm{T}>i-1\}}(1 + x_{\mathrm{T}}) + \mathbb{1}_{\{\mathrm{T}\le i-1\}}(x_{i-1} + x_i)] - (x_i - x_{i+1}) \\ &- \lambda(y_{i-1} - y_i)[\mathbb{1}_{\{\mathrm{T}>i-1\}}(1 + y_{\mathrm{T}}) + \mathbb{1}_{\{\mathrm{T}\le i-1\}}(y_{i-1} + y_i)] + (y_i - y_{i+1})|,\end{aligned}$$

$$\begin{aligned}&\le 2\sum_{i=0}^{\infty} |x_i - y_i| + 2\lambda \sum_{i=0}^{\infty} [\mathbb{1}_{\{\mathrm{T}>i-1\}}(|x_i - y_i| + |x_i x_{\mathrm{T}} - y_i y_{\mathrm{T}}|) \\ &\quad + \mathbb{1}_{\{\mathrm{T}\le i-1\}}|x_i^2 - y_i^2|], \\ &\le 2\sum_{i=0}^{\infty} |x_i - y_i| + 2\lambda \sum_{i=0}^{\infty} [\mathbb{1}_{\{\mathrm{T}>i-1\}}(|x_i - y_i| + |x_i^2 - y_i^2|) \\ &\quad + \mathbb{1}_{\{\mathrm{T}\le i-1\}}|x_i^2 - y_i^2|], \\ &\le \sum_{i=0}^{\infty} (2 + \mathbb{1}_{\{\mathrm{T}>i-1\}}6\lambda + \mathbb{1}_{\{\mathrm{T}\le i-1\}}4\lambda)|x_i - y_i|.\end{aligned}$$

Note that $0 \le x_i,\, y_i \le 1$. □

4.2 Analysis of the Fixed Point

Proof of Theorem 3 & Lemma 1. Recall that the fixed point of the ODE (3) associated with the threshold supermarket model satisfies the following algebraic recursion equations:

$$\begin{cases} s_{i+1} = \begin{cases} s_i - \lambda(s_{i-1} - s_i)(1 + s_{\mathrm{T}}) & \text{if } T > i - 1 \\ s_i - \lambda(s_{i-1}^2 - s_i^2) & \text{if } T \le i - 1 \end{cases} & \text{for } i \ge 2, \\ s_0 = 1, \\ s_1 = \lambda. \end{cases}$$

4.2.1 Step 1: Finding s_i for $i \leq T+1$

We begin by solving the recursion for $T \leq i-1$, assuming s_T is given.

Note that we have a linear homogeneous recurrence relation with constant coefficients of the form $s_{n+1} = c_1 s_n + c_2 s_{n-1}$ where c_1 and $c_2 \in \mathbb{R}$ and $c_2 \neq 0$. For such a relation, suppose $x^2 = c_1 x + c_2$ has two distinct roots r_1 and r_2. Then we know $\{s_{n+1}\}$ is a solution of the recursion $s_{n+1} = c_1 s_n + c_2 s_{n-1}$ if and only if $s_{n+1} = \alpha_1 r_1^n + \alpha_2 r_2^n$ for $n \in \mathbb{N}$ and some constants α_1, α_2.

So, from our recursive relation $(s_{i+1} = (1+\lambda(1+s_{\mathrm{T}}))s_i - (\lambda(1+s_{\mathrm{T}}))s_{i-1})$, we know that $c_1 = 1+\lambda(1+s_{\mathrm{T}})$ and $c_2 = -\lambda(1+s_{\mathrm{T}})$. Then we examine the associated quadratic equation:

$$x^2 - (1+(\lambda(1+s_{\mathrm{T}}))x + \lambda(1+s_{\mathrm{T}}) = 0,$$
$$x^2 - x - x\lambda(1+s_{\mathrm{T}}) + \lambda(1+s_{\mathrm{T}}) = 0,$$
$$x(x-1) - \lambda(1+s_{\mathrm{T}})(x-1) = 0,$$
$$[x - \lambda(1+s_{\mathrm{T}})][x-1] = 0,$$

And so, $r_1 = \lambda(1+s_{\mathrm{T}})$ and $r_2 = 1$.

(Note that we now assume that $\lambda(1+s_T) \neq 1$. In what follows s_T will be determined by an implicit equation involving λ– we will later investigate for what values of λ this condition is satisfied.)

We must now find α_1 and α_2 such that $s_i = \alpha_1(\lambda^i(1+s_{\mathrm{T}})^i) + \alpha_2$.
We know: $s_0 = 1 = \alpha_1 + \alpha_2$, and $s_1 = \lambda = \alpha_1(\lambda(1+s_{\mathrm{T}})) + \alpha_2$.
By substitution:

$$\lambda = \alpha_1(\lambda(1+s_{\mathrm{T}})) + 1 - \alpha_1,$$
$$\alpha_1(1 - \lambda(1+s_{\mathrm{T}})) = 1 - \lambda,$$
$$\alpha_1 = \tfrac{1-\lambda}{1-\lambda(1+s_{\mathrm{T}})} \text{ and } \alpha_2 = \tfrac{\lambda-\lambda(1+s_{\mathrm{T}})}{1-\lambda(1+s_{\mathrm{T}})}.$$

We have now solved the first part of our recursion:

$$\boxed{s_i = \left[\frac{1-\lambda}{1-\lambda(1+s_{\mathrm{T}})}\right]\lambda^i(1+s_{\mathrm{T}})^i + \left[\frac{\lambda-\lambda(1+s_{\mathrm{T}})}{1-\lambda(1+s_{\mathrm{T}})}\right] \text{ for } 2 \leq i \leq T+1}.$$

4.2.2 Step 2: Finding s_{T}

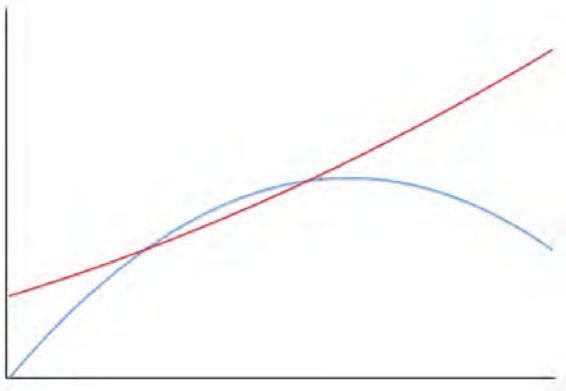
We can now use this equation to solve for s_{T}. By substituting $i = T$ in the above boxed expression for s_i, we see that s_{T} must satisfy:

$$(\lambda-1)\lambda^T(1+s_{\mathrm{T}})^T - \lambda s_{\mathrm{T}}^2 + s_{\mathrm{T}} = 0.$$

Although we cannot explicitly solve this for s_T for a general T, we can find important properties that reveal how the equation behaves by arranging it in the following convenient way:

$$(1 + s_T)^T = \frac{s_T - \lambda s_T^2}{\lambda^T(1 - \lambda)}.$$

As in equation (4), we define H_1 to be the left-hand side and H_2 to be the right-hand side. It is intuitively obvious that the H_1 is increasing and convex in the interval [0, 1], and the H_2 is an inverted parabola. The graph of the H_1 and H_2 looks generally like this:



Now that we have this insight, we return to the analytics and find that:

1. At $s_T = 0$:
   i. $H_1(0) > H_2(0)$.
      Proof: $H_1(0) = 1 > H_2(0) = 0$.

   ii. $H_2'(0) > H_1'(0)$
      Proof: $H_2'(0) = \frac{1-2\lambda s_T}{\lambda^T(1-\lambda)} = \frac{1}{\lambda^T(1-\lambda)}$. $H_1'(0) = T(1 + s_T)^{T-1} = T$. Note that for $0 < \lambda < 1$, $\lambda^T(1 - \lambda) > 0$. So, proving that $H_2' > H_1'$ is equivalent to proving that $\frac{1}{T} > \lambda^T(1 - \lambda)$. And so, if $\frac{1}{T}$ always exceeds the maximum of $\lambda^T(1 - \lambda)$, then we have completed our task.

Finding the max: $\frac{d[\lambda^T(1-\lambda)]}{d\lambda} = -\lambda^T + (1 - \lambda)T\lambda^{T-1} = 0$,
$T(1 - \lambda)\lambda^{T-1} = \lambda^T$,
$T - \lambda T = \lambda$,
$\lambda(1 + T) = T$,
$\lambda^{\max} = \frac{T}{T+1}$.

Plugging $\lambda^{\max}$ in, the maximum is $\left[\frac{T}{T+1}\right]^T \left[\frac{1}{T+1}\right]$. So, if $\frac{1}{T} > \left[\frac{T}{T+1}\right]^T \left[\frac{1}{T+1}\right]$, we have finished. Because $\frac{T}{T+1} < 1$ and $T \geq 2$, we know $\left[\frac{T}{T+1}\right]^T < 1$ as well. Additionally, because

$T+1$ is larger than T, we know $\frac{1}{T+1} < \frac{1}{T}$. Therefore, we know $\left[\frac{T}{T+1}\right]^T \left[\frac{1}{T+1}\right]$ is the product of a quantity that is less than $\frac{1}{T}$ and a quantity that is less than 1, thereby making it necessarily less than $\frac{1}{T}$. Therefore, $\frac{1}{T}$ is clearly greater than $\lambda^T(1-\lambda)$. Our task is complete.

iii. $H_1'(0), H_2'(0) > 0$.

Proof: $H_1'(0) = T(1+s_{\mathrm{T}})^{T-1} = T$. Because $T \geq 2$, it is obvious that $T > 0$ and therefore, $H_1'(0) > 0$.

$H_2' = \frac{1-2\lambda s_{\mathrm{T}}}{\lambda^T(1-\lambda)} = \frac{1}{\lambda^T(1-\lambda)}$. Because $0 < \lambda < 1$, it is clear that both λ^T and $(1-\lambda)$ are positive. Thus, both the denominator and numerator are positive. Because the quotient of two positive numbers is always positive, $H_2'(0) > 0$.

2. H_1 begins at $H_1(0) = 1$ and ends at $H_1(1) = 2^T$ and is positive, increasing, and convex for all s_{T} in the interval [0, 1].

i. $H_1 > 0$ for all s_{T}.

Proof: Because $s_{\mathrm{T}} \in [0,\ 1]$, we know $H_1 \in [1,\ 2^T] > 0$.

ii. $H_1' > 0$ for all s_{T}.

Proof: $H_1' = T(1+s_{\mathrm{T}})^{T-1}$. Because $T \geq 2$, it is obvious that $T > 0$, and because $s_T \in [0,\ 1]$, it is obvious that $(1+s_{\mathrm{T}}) \in [1,\ 2]$ and therefore $(1+s_{\mathrm{T}})^{T-1} > 0$. Because the product of two positive quantities is always positive, $H_1' > 0$ for all s_{T}.

iii. $H_1'' > 0$ for s_{T}.

Proof: $H_1'' = T(T-1)(1+s_{\mathrm{T}})^{T-2}$. Based on the same reasoning as above, we know that $T \geq 2$, $(T-1) \geq 1$, and $(1+s_{\mathrm{T}})^{T-2} \geq 1$. Again, because the product of only positive numbers is always positive, $H_1'' > 0$ for all s_{T}.

3. H_2 begins at $H_2(0) = 0$, increases from $s_{\mathrm{T}} = 0$ to some critical point $s_{\mathrm{T}}^{\max}$, decreases thereafter, and ends at $H_2(1) = \frac{1}{\lambda^T}$. Additionally, H_2 is convex for all s_{T}.

i. $H_2' > 0$ for $s_{\mathrm{T}} \in [0,\ s_{\mathrm{T}}^{\max})$.

Proof: First, we must find the critical point $s_{\mathrm{T}}^{\max}$.

At the maximum, $H_2' = \frac{1-2\lambda s_{\mathrm{T}}}{\lambda^T(1-\lambda)} = 0$. So we must find $1 - 2\lambda s_{\mathrm{T}} = 0$, which occurs when $s_{\mathrm{T}}^{\max} = \frac{1}{2\lambda}$.

Now let us examine H_2' for $s_{\mathrm{T}} \in [0,\ s_{\mathrm{T}}^{\max})$: $H_2' = \frac{1-2\lambda s_{\mathrm{T}}}{\lambda^T(1-\lambda)}$. Because $0 < \lambda < 1$, it is clear that $\lambda^T(1-\lambda) > 0$. Therefore, H_2' is positive when $1 - 2\lambda s_{\mathrm{T}} > 0$. When $s_{\mathrm{T}} < \frac{1}{2\lambda}$, $1 > 2\lambda s_{\mathrm{T}}$ and therefore $H_2' > 0$ in the interval $[0,\ s_{\mathrm{T}}^{\max})$.

ii. $H_2' < 0$ for $s_{\mathrm{T}} \in (\ s_{\mathrm{T}}^{\max},\ 1]$.

Proof: $H_2' = \frac{1-2\lambda s_{\mathrm{T}}}{\lambda^T(1-\lambda)}$. By the same reasoning as above, because the denominator is always positive, H_2' is only negative if $1 - 2\lambda s_{\mathrm{T}} < 0$. For $s_{\mathrm{T}} > \frac{1}{2\lambda}$, it is obvious that this inequality holds and therefore, $H_2' < 0$ for $s_{\mathrm{T}} \in (\ s_{\mathrm{T}}^{\max},\ 1]$.

iii. $H_2'' < 0$ for all s_{T}.

Proof: $H_2'' = \frac{-2\lambda}{\lambda^T(1-\lambda)}$. Because $0 < \lambda < 1$, it is clear that $-2\lambda < 0$ and $\lambda^T(1-\lambda) > 0$. Because a negative quantity divided by a positive quantity is always negative, it is clear that $H_2'' < 0$ for all s_{T}.

All of these properties combined tell us that: if at some $s_{T^*} > 0$, $H_2 \geq H_1$, then the H_1 and H_2 at least once and at most twice in the interval [0, 1]. Furthermore, we know:

(A) When $H_1 > H_2$, $H_2' > H_1'$, and $H_1', H_2' > 0$, there have been zero intersections.

(B) When $H_2 > H_1$, there has been exactly one intersection.

(C) When $H_1 > H_2$, $H_2' < H_1'$, and there has been at least one point below s_{T} at which $H_1 \leq H_2$, there have ben exactly two intersections.

So, we come to the following conclusions:

As proven in (1), at $s_{\mathrm{T}} = 0$, there have been exactly zero intersections.
Now, we will examine the equation at $s_{\mathrm{T}} = 1$:

I. If $\lambda < \frac{1}{2}$: $H_2 > H_1$ and therefore, in the interval [0, 1], there is exactly one intersection.

Proof: $H_1(1) = 2^T$. $H_2(1) = \frac{1-\lambda}{\lambda^T(1-\lambda)} = \frac{1}{\lambda^T}$. We must show that $2^T < \frac{1}{\lambda^T}$, which is the same as showing $(2\lambda)^T < 1$. Because $\lambda < \frac{1}{2}$, we know $2\lambda < 1$ and therefore,$(2\lambda)^T < 1^T = 1$. So, the inequality holds.

II. If $\lambda > \frac{1}{2}$: $H_1 > H_2$, $H_2' < H_1'$, and there has been at least one point below s_{T} at which $H_1 \leq H_2$ and therefore, in the interval [0, 1], there have been exactly two intersections.

Proof: i. $H_1(1) = 2^T$. $H_2(1) = \frac{1-\lambda}{\lambda^T(1-\lambda)} = \frac{1}{\lambda^T}$. We must show that $2^T > \frac{1}{\lambda^T}$, which is the same as showing $(2\lambda)^T > 1$. Because $\lambda > \frac{1}{2}$, we know $2\lambda > 1$ and therefore, $(2\lambda)^T > 1^T = 1$. So, the inequality holds.

ii. $H_2' = \frac{1-2\lambda s_{\mathrm{T}}}{\lambda^T(1-\lambda)} = \frac{1-2\lambda}{\lambda^T(1-\lambda)}$. $H_1' = T(1 + s_{\mathrm{T}})^{T-1} = T(2)^{T-1}$. Because $T \geq 2$, $H_1' \geq 4$, which is positive. Because $\lambda > \frac{1}{2}$, it is clear that $H_2' < 0$. Therefore, $H_1' > H_2'$.

iii. $\overline{s}_{\mathrm{T}} = \frac{\lambda^3 - 2\lambda^2 + 1}{\lambda^2 - \lambda^3 + \lambda} < 1$, and we know that at this point $\overline{s}_{\mathrm{T}}$, $H_1 = H_2$ (see Section 4.2.3). Therefore, there has been at least one point below s_{T} at which $H_1 \leq H_2$.

III. If $\lambda = \frac{1}{2}$: $H_1 = H_2$, $H_1' > H_2'$, and there has been at least one point below $s_{\mathbf{T}}$ at which $H_1 \leq H_2$ and therefore, in the interval [0, 1], there have been exactly two intersections. One intersection at $s_{\mathbf{T}} = 1$ and one in the interval [0,1).

Proof: i. $H_1(1) = 2^T$ and $H_2(1) = \frac{1-\frac{1}{2}}{\frac{1}{2}^T} = 2^T$.

ii. $H_2' = \frac{1-1}{\frac{1}{2}^{T+1}} = 0$ and $H_1' = T(2)^{T-1} > 0$. Therefore, $H_1' > H_2'$.

iii. Because $H_1(0) > H_2(0)$, $H_1'(0) < H_2'(0)$, $H_2(1) = H_1(1)$, $H_2'(1) = 0$, and $H_1'(1) > 0$, there must be one point below $s_{\mathrm{T}} = 1$ at which the H_1 and H_2 intersect.

4.2.3 Step 3: A Closer Examination of $\lambda > \frac{1}{2}$ (Finding the T-Independent Root)

To better understand the case when $\lambda > \frac{1}{2}$, we begin by looking at the case that is easy to analyze ($T = 2$). Plugging in $T = 2$, we are left with a simple quadratic equation:

$$(\lambda^2 - \lambda^3 + \lambda)s_{\mathrm{T}}^2 \; + \; (2\lambda^2 - 2\lambda^3 - 1)s_{\mathrm{T}} \; + \; (\lambda^2 - \lambda^3) \; = \; 0.$$

By using the quadratic formula, we find the following two roots:

$$\begin{aligned} r_1 \; &= \; \tfrac{\lambda^3 - 2\lambda^2 + 1}{\lambda^2 - \lambda^3 + \lambda} \; = \; \tfrac{1-\lambda}{\lambda}, \\ r_2 \; &= \; \tfrac{\lambda^2}{1+\lambda-\lambda^2}. \end{aligned}$$

By plugging each of these two roots back into the equation with general T, we find that r_1, which we define as $\overline{s}_{\mathrm{T}}$, is in fact a root for all T.

Proof: By plugging this $\overline{s}_{\mathrm{T}}$ into $(\lambda - 1)\lambda^T(1 + s_{\mathrm{T}})^T \; - \; \lambda s_{\mathrm{T}}^2 \; + \; s_{\mathrm{T}} \; = \; 0$, we find:

$$\lambda^T(\lambda - 1)\left[\frac{\lambda^2 - \lambda^3 + \lambda + \lambda^3 - 2\lambda^2 + 1}{\lambda(\lambda - \lambda^2 + 1)}\right]^T - \lambda\left[\frac{\lambda^6 - 4\lambda^5 + 4\lambda^4 + 2\lambda^3 - 4\lambda^2 + 1}{\lambda(\lambda^5 - 2\lambda^4 - \lambda^3 + 2\lambda^2 + \lambda)}\right]$$

$$+ \left[\frac{\lambda^3 - 2\lambda^2 + 1}{\lambda(\lambda - \lambda^2 + 1)}\right],$$

$$= (\lambda - 1) \; - \; \left[\frac{\lambda^6 - 4\lambda^5 + 4\lambda^4 + 2\lambda^3 - 4\lambda^2 + 1}{\lambda^5 - 2\lambda^4 - \lambda^3 + 2\lambda^2 + \lambda}\right] \; + \; \left[\frac{\lambda^3 - 2\lambda^2 + 1}{\lambda(\lambda - \lambda^2 + 1)}\right],$$

$$= \left[\frac{\lambda^6 - 4\lambda^5 + 4\lambda^4 + 2\lambda^3 - 4\lambda^2 + 1}{\lambda^5 - 2\lambda^4 - \lambda^3 + 2\lambda^2 + \lambda}\right] \; + \; \left[\frac{\lambda^4 - 3\lambda^3 + 2\lambda^2 + \lambda - 1}{\lambda - \lambda^2 + 1}\right],$$

$$= \left[\frac{(\lambda^6 - 4\lambda^5 + 4\lambda^4 + 2\lambda^3 - 4\lambda^2 + 1)(\lambda - \lambda^2 + 1)}{(\lambda^5 - 2\lambda^4 - \lambda^3 + 2\lambda^2 + \lambda)(\lambda - \lambda^2 + 1)}\right]$$

$$+ \; \left[\frac{(\lambda^4 - 3\lambda^3 + 2\lambda^2 + \lambda - 1)(\lambda^5 - 2\lambda^4 - \lambda^3 + 2\lambda^2 + \lambda)}{(\lambda^5 - 2\lambda^4 - \lambda^3 + 2\lambda^2 + \lambda)(\lambda - \lambda^2 + 1)}\right],$$

$$= \left[\lambda^8 - 5\lambda^7 + 7\lambda^6 + 2\lambda^5 - 10\lambda^4 + 2\lambda^3 + 5\lambda^2 - \lambda - 1\right]$$

$$- \left[\lambda^8 - 5\lambda^7 + 7\lambda^6 + 2\lambda^5 - 10\lambda^4 + 2\lambda^3 + 5\lambda^2 - \lambda - 1\right],$$

$$= 0.$$

However, the root $\overline{s}_{\mathrm{T}}$ is not a valid root since $\lambda(1+\overline{s}_{\mathrm{T}}) = 1$, which violates the condition found earlier that $\lambda(1+\overline{s}_{\mathrm{T}}) \neq 1$. Therefore, the root we are interested in should satisfy the algebraic identity:

$$\frac{(\lambda-1)\lambda^T(1+s_{\mathrm{T}})^T - \lambda s_{\mathrm{T}}^2 + s_{\mathrm{T}}}{s_{\mathrm{T}} - \overline{s}_{\mathrm{T}}} \; = \; 0.$$

But, in what interval will the unique solution lie? The numerics show that for some values of T, $\overline{s}_{\mathrm{T}}$ is the larger of the two roots, while for other values of T, $\overline{s}_{\mathrm{T}}$ is the smaller of the two roots. So, is there a simple expression to determine for which T, the stable root s_{T^*} will be in the interval $[0,\, \overline{s}_{\mathrm{T}})$ and for which in the interval $(\overline{s}_{\mathrm{T}},\, 1]$? The answer lies in the derivative.

$$\begin{aligned} \text{If we define: } G(T,\, s_{\mathrm{T}}) \; &= \; (\lambda - 1)\lambda^T(1 + s_{\mathrm{T}})^T \; - \; \lambda s_{\mathrm{T}}^2 \; + \; s_{\mathrm{T}}, \\ \text{then, } G'(T,\, s_{\mathrm{T}}) \; &= \; \lambda^T(1 - \lambda)T(1 + s_{\mathrm{T}})^{T-1} \; + \; 2\lambda s_{\mathrm{T}} \; - \; 1. \end{aligned}$$

By examining G' at the point $\overline{s}_{\mathrm{T}}$, we find that $G'(T,\, \overline{s}_{\mathrm{T}}) \; > \; 0$ when $T > \frac{2\lambda-1}{\lambda(1-\lambda)}$ and $G'(T,\, \overline{s}_{\mathrm{T}}) < 0$ when $T \frac{2\lambda-1}{\lambda(1-\lambda)}$. Thus, we now have our answer. In order to solve for the

unique point s_{T^*} which our system converges to when $\lambda > \frac{1}{2}$, we must find the solution to:

$$\frac{(\lambda-1)\lambda^T(1+s_{\mathrm{T}})^T \;-\; \lambda s_{\mathrm{T}}^2 \;+\; s_{\mathrm{T}}}{s_{\mathrm{T}} \;-\; \left[\frac{\lambda^3-2\lambda^2+1}{\lambda^2-\lambda^3+\lambda}\right]} \;=\; 0, \text{ in the interval}$$

$$i)\ [0,\ \overline{s}_{\mathrm{T}}) \text{ if } T > \frac{2\lambda-1}{\lambda(1-\lambda)},$$

$$ii)\ (\overline{s}_{\mathrm{T}},\ 1] \text{ if } T < \frac{2\lambda-1}{\lambda(1-\lambda)}.$$

Therefore, we now have an algorithm for finding the unique point (s_{T^*}) our system converges to for all λ:

1) If $\lambda < \frac{1}{2}$, simply solve for s_{T^*} such that $(\lambda-1)\lambda^T(1+s_{\mathrm{T}})^T \;-\; \lambda s_{\mathrm{T}}^2 \;+\; s_{\mathrm{T}} = 0$ in the interval $[0,\ 1]$.

2) If $\lambda = \frac{1}{2}$, solve for s_{T^*} such that $\dfrac{(\lambda-1)\lambda^T(1+s_{\mathrm{T}})^T \;-\; \lambda s_{\mathrm{T}}^2 \;+\; s_{\mathrm{T}}}{s_{\mathrm{T}} \;-\; \overline{s}_{\mathrm{T}}} = 0$ in the interval $[0,\ 1)$.

3) If $\lambda > \frac{1}{2}$, solve for s_{T^*} such that $\dfrac{(\lambda-1)\lambda^T(1+s_{\mathrm{T}})^T \;-\; \lambda s_{\mathrm{T}}^2 \;+\; s_{\mathrm{T}}}{s_{\mathrm{T}} \;-\; \overline{s}_{\mathrm{T}}} = 0$

i) in the interval $[0,\ \overline{s}_{\mathrm{T}})$ if $T > \dfrac{2\lambda-1}{\lambda(1-\lambda)}$,

ii) in the interval $(\overline{s}_{\mathrm{T}},\ 1]$ if $T < \dfrac{2\lambda-1}{\lambda(1-\lambda)}$.

This proves Lemma 1.

4.2.4 Step 4: Finding s_i for $i > T+1$

Now that we know s_{T} and s_i for $i \leq T+1$, we can find s_i for $i > T+1$. Recall that we will be solving the following recursive relation with initial conditions s_{T} and $s_{\mathrm{T}+1}$:

$$s_{i+1} \;=\; s_i \;-\; \lambda(s_{i-1}^2 \;-\; s_i^2) \quad \text{for } i \geq \; T \;+\; 1.$$

We will begin by proving a simple and convenient relation between s_{T} and $s_{\mathrm{T}+1}$ through taking advantage of a telescoping sum.

From $s_{i+1} - s_i = \lambda(s_{i-1} - s_i)(1 + s_{\mathrm{T}})$ for $i < T + 1$, we know that:

$$\sum_{i=1}^{T}(s_{i+1} - s_i) = (s_2 - s_1) + (s_3 - s_2) + (s_4 - s_3) + \ldots + (s_{\mathrm{T}+1} - s_{\mathrm{T}}) = s_{\mathrm{T}+1} - s_1 = s_{\mathrm{T}+1} - \lambda.$$

And $\sum_{i=1}^{T} \lambda(s_{i-1} - s_i)(1 + s_{\mathrm{T}}) = \lambda(1 + s_{\mathrm{T}})[(s_1 - s_0) + (s_2 - s_1) + (s_3 - s_2) + \ldots + (s_{\mathrm{T}} - s_{\mathrm{T-1}})]$,

$= \lambda(1 + s_{\mathrm{T}})(s_{\mathrm{T}} - s_0) = \lambda(1 + s_{\mathrm{T}})(s_{\mathrm{T}} - 1)$.

Therefore, $s_{\mathrm{T}+1} - \lambda = \lambda(s_{\mathrm{T}} + 1)(s_{\mathrm{T}} - 1) = \lambda(s_{\mathrm{T}}^2 - 1)$, and $\boxed{s_{\mathrm{T}+1} = \lambda s_{\mathrm{T}}^2}$.

We now return to the current recursion. In a similar fashion, we take advantage of a telescoping sum (from $T + 1$ to an integer N that is sufficiently large) to simplify the expression in question.

From $s_{i+1} - s_i = \lambda(s_{i-1}^2 - s_i^2)$ for $i \geq T + 1$, we know that:

$$\sum_{i=T+1}^{N} (s_{i+1} - s_i) = (s_{\mathrm{T}+2} - s_{\mathrm{T}+1}) + (s_{\mathrm{T}+3} - s_{\mathrm{T}+2}) + \ldots + (s_{\mathrm{N}+1} - s_{\mathrm{N}}) = s_{\mathrm{N}+1} - s_{\mathrm{T}+1}.$$

And $\sum_{T+1}^{N} \lambda(s_{i-1}^2 - s_i^2) = \lambda[(s_{\mathrm{T}+1}^2 - s_{\mathrm{T}}^2) + (s_{\mathrm{T}+2}^2 - s_{\mathrm{T}+1}^2) + \ldots + (s_{\mathrm{N}}^2 - s_{\mathrm{N-1}}^2)] = \lambda(s_{\mathrm{N}}^2 - s_{\mathrm{T}}^2)$.

Therefore, $s_{\mathrm{N}+1} = (s_{\mathrm{T}+1} - \lambda s_{\mathrm{T}}^2) + \lambda s_{\mathrm{N}}^2$.

And by plugging in $s_{\mathrm{T}+1} = \lambda s_{\mathrm{T}}^2$,we find that $\boxed{s_{\mathrm{N}+1} = \lambda s_{\mathrm{N}}^2}$.

To simplify this recursion, we will transform it by taking the logarithm base λ and substituting in $a_n = log_\lambda s_N$. Thus, we come to the following linear homogeneous recurrence relation with constant coefficients, which we can solve using the same method as Step 1.

$$a_{n+2} = 3a_{n+1} - 2a_n.$$

Recall that we first must find the roots of the following equation: $x^2 - 3x + 2 = 0$. The resulting roots $r_1 = 2$ and $r_2 = 1$ lead us to the following relationship: $a_i = \alpha_1 2^i + \alpha_2$. And substituting back in, we find that $log_\lambda s_i = \alpha_1 2^i + \alpha_2$.

To solve for α_1 and α_2, we use the initial conditions and the fact that $s_{\mathrm{T}+1} = \lambda s_{\mathrm{T}}^2$, which give us the following three equations:

$$\begin{cases} log_\lambda s_{\mathrm{T}} = \alpha_1 2^T + \alpha_2, \\ log_\lambda s_{\mathrm{T}+1} = \alpha_1 2^{T+1} + \alpha_2, \\ log_\lambda s_{\mathrm{T}+1} = 1 + 2log_\lambda s_{\mathrm{T}}. \end{cases}$$

Through substitution, we see that $\alpha_1 = \frac{1+log_\lambda s_T}{2^T}$ and $\alpha_2 = -1$, and therefore $log_\lambda s_i = (1 + log_\lambda s_T)2^{i-T} - 1$. By taking the exponential of both sides, we come to our answer:

$$\boxed{s_i \;=\; s_T(\lambda s_T)^{2^{i-T}-1}}.$$

4.2.5 Conclusion

We have now proven equation (5). Given s_{T^*}, the fixed point of our system is:

$$\boxed{\begin{aligned} s_0 &= 1, \\ s_1 &= \lambda, \\ s_i &= \begin{cases} \left[\dfrac{1 \;-\; \lambda}{1 \;-\; \lambda(1+s_{T^*})}\right]\lambda^i(1+s_{T^*})^i \;+\; \left[\dfrac{\lambda \;-\; \lambda(1+s_{T^*})}{1 \;-\; \lambda(1+s_{T^*})}\right] & \text{for } 2 \le i \le T+1, \\ s_{T^*}(\lambda s_{T^*})^{2^{i-T}-1} & \text{for } i > T+1. \end{cases} \end{aligned}}$$

Using arguments similar to those used in [1], it can be shown that the limiting stationary distribution $\{\pi_i\}_{i=0}^{\infty}$ is equal to this computed fixed point $\{s_i\}_{i=0}^{\infty}$. □

5 The Threshold Supermarket Game

5.1 Description of the Model

The classic load balancing problem traditionally focuses on models with a central dispatcher that distributes load across the system. Often times, however, customers themselves must decide which queue to join, and most often, they prefer queues of shorter length. However, because sampling queues is costly, customers must find the right balance between probing enough queues to find one that is sufficiently short but not so many that the cost from sampling outweighs the benefit. Because an arriving customer's expected queue length depends on the choices of all preceding customers' choices, this problem is a game among customers.

The classic supermarket game, with Poisson arrivals to n parallel FIFO queues and i.i.d. exponential service times with mean $\mu = 1$, has previously been analyzed by Xu and Hajek [16] in their paper titled "The Supermarket Game." A summary of their paper can be found in Section 5.3. We aim to expand upon this and look at a variation of the supermarket game in which d, the number of queues sampled by each customer, is dependent on the length of the initial queue observed and some threshold T, following the same dynamics defined in equation (2). Upon entering the system, each customer arrives to her dedicated queue i. If the customer's dedicated queue has length q which is greater than or equal to some threshold T, an alternate queue is sampled, and the customer joins the shorter

queue. If the queue length q is less than the threshold T, the customer simply joins this queue, bypassing the extra work and cost of an additional search. Note that the dynamics of this system are equivalent to those of the threshold supermarket model described above (in which the initial queue is selected uniformly at random upon arrival); however, this set-up proves more relevant to this associated game.

Customers must choose a strategy to determine what threshold T they will use when arriving to the system. A customer can devise a pure strategy or a mixed strategy. A pure strategy for customer i corresponds to a sampling strategy $d(q,\, T_i)$, for some deterministic threshold choice $T_i \in \{0, 1, 2, 3, \ldots\}$. Alternatively, a customer can devise a mixed strategy, which assigns a certain probability to each possible pure strategy. In other words, a customer can randomize over the set of pure strategies in order to choose a threshold value. In this case, customers are assumed to use the strategy μ_i— namely, they randomize over a set $\mathcal{T}$ of a finite number of thresholds, where the probability that player i uses the sampling strategy $d(q,\, T_i)$ with threshold T_i is $\mu_i(T_i)$. We will only consider pure strategies.

Note that when all customers use the same pure strategy, namely the sampling strategy $d(q,\, T_i)$ for some deterministic $T_i \in \mathbb{N}$, the game will have a limiting stationary expected queue length distribution that corresponds to that of our basic threshold model described in Section 4. Additionally, note that if all customers use the optimal pure strategy threshold T^* found in Section 3.3, the resulting cost coincides with a socially optimal equilibrium. That is, when all customers follow strategy T^*, the resulting equilibrium is the solution to what is called the "social planner problem" in economics. As the name implies, the "social planner problem" seeks to find the equilibrium that is collectively the best for all players. In this previous analysis of the socially optimal equilibrium, we assumed the existence of a centralized social planner who imposed a universal optimal threshold on the system. We now shift our focus to the more complicated decentralized setting in which each customer follows her own strategy for choosing a threshold.

We consider the situation where a customer arriving to queue i uses the pure strategy associated with the threshold T_i, while all other customers use the pure strategy corresponding to a (possibly) different threshold T_{-i}. Let $\{s_i(T_{-i})\}$ be the limit stationary expected queue length distribution corresponding to the threshold model with threshold T_{-i}. Then the distribution of the remaining queues, as observed by any customer i, will be $\{s_i(T_{-i})\}$, and this will determine the distribution of the queues sampled by customers joining queue i. Also, let $\tilde{s}_i(T_i,\, T_{-i})$ be the stationary distribution of queue i when customers arriving at queue i use the threshold T_i, and customers arriving at all other queues use threshold T_{-i}. Then the cost to customers arriving to queue i is given by

$$\begin{aligned} C(T_i,\, T_{-i}) &= \mathbb{E}[\text{queue length}] \,+ c_s \mathbb{E}[\text{number of extra searches}], \\ &= \sum_{i=1}^{\infty} \tilde{s}_i(T_i,\, T_{-i}) \,+\, c_s s_{T_i}(T_{-i}). \end{aligned} \tag{7}$$

For simplicity, assume we are given lower and upper thresholds $0 \leq T_{min} \leq T_{max} < \infty$, such that the thresholds T_i and T_{-i} must lie in $[T_{min},\, T_{max}]$.

Let T^{NE} denote the pure strategy Nash equilibrium of our threshold supermarket game. A threshold T^{NE} is a pure strategy Nash equilibrium if

$$C(T^{\text{NE}}, T^{\text{NE}}) \;\leq\; C(T_i, T^{\text{NE}}), \text{ for all } T_i \text{ in } \{T_{min},\, ...\, T_{max}\}.$$

5.2 Main Results

Recall that the socially optimal strategy results in an equilibrium that makes all players collectively better off, and the Nash equilibrium is that which is reached when each player acts selfishly, choosing a strategy that makes only one's self better off, regardless of the effect on others. It is obvious that policymakers desire a system to reach the socially optimal equilibrium. Accordingly, policymakers, often times, put incentives, such as taxes or subsidies, into place to tempt players to behave in such a way that will allow the system to reach the socially optimal equilibrium.

With this in mind, we ask: Given a system's cost of sampling c_s, is it possible to reach the social optimum in this decentralized setting by providing incentives to customers? We aim to find some $\widehat{c_s}$, where $|\widehat{c_s} - c_s|$ will serve as either a subsidy or tax, such that when imposed, there exists a Nash equilibrium strategy, which we will call $\widehat{T}^{\text{NE}}(\widehat{c_s})$, that leads to an equilibrium average total cost $C(\widehat{T}^{\text{NE}},\, \widehat{T}^{\text{NE}})$ equal to that of the social optimum. In other words, we seek some $\widehat{c_s}$ which leads customers to self-organize in such a way that allows the system to reach its social optimum.

To complete this analysis, we must first understand the distribution $\tilde{s}_i$. It is easy to see that $\tilde{s}_i$ is the stationary distribution associated with a birth-death process and be represented by the following recursion:

$$\begin{cases} \tilde{s}_0 \;=\; 1, \\ \tilde{s}_1 \;=\; \lambda, \\ \tilde{s}_i(T_i,\, T_{-i}) \;-\; \tilde{s}_{i+1}(T_i,\, T_{-i}) \;=\; \begin{cases} \lambda(s_{i-1}(T_{-i}) - s_i(T_{-i}))(1 + s_{T_i}(T_{-i})), & \text{for } 1 \leq i \leq T_i, \\ \lambda(s^2_{i-1}(T_{-i}) - s^2_i(T_{-i})), & \text{for } i > T_i. \end{cases} \end{cases}$$

Note that when $T_i = T_{-i} = T$, this reduces to the algebraic equations (3) that characterize the fixed point for the supermarket model with threshold T. (But they are different when $T_i \neq T_{-i}$.)

To find $\tilde{s}_i$, one must simply take the sum of the above quantity from i to infinity. By taking advantage of the resulting telescoping sum, the distribution for $i > 1$ simplifies to:

$$\tilde{s}_i \;=\; \begin{cases} \lambda(1 + s_{T_i})(s_{i-1} - s_{T_i}) \;+ \lambda s^2_{T_i} & \text{for } 1 \leq i \leq T_i, \\ \lambda s^2_{i-1} & \text{for } i > T_i. \end{cases} \tag{8}$$

Using this equation, for a given cost c_s, we can now find $\widehat{c_s}$ which, when imposed, leads to a Nash equilibrium total average cost which coincides with the socially optimal total cost (or $C(\widehat{T}^{\mathrm{NE}}, \widehat{T}^{\mathrm{NE}}) = C(T^*, T^*)$). The following table displays what we find.

Table 5: **Social Optimum Costs Coinciding with Nash Equilibrium Costs**

λ	c_s	T^*	$\widehat{c_s}$	$\widehat{T}^{\mathrm{NE}}$	*Equil. Cost*
0.40	0.1	1	0.1	1	0.5056
	0.2	1	0.2	1	0.5456
	0.3	2	0.3	2	0.5744
	0.4	2	0.4	2	0.5873
	0.5	2	0.5	2	0.6002
	0.7	3	0.7	3	0.6259
	0.8	3	0.8	3	0.6303
	0.9	3	0.9	3	0.6348
	1.1	4	1.2	3	0.6477
	1.3	4	1.3	4	0.6510
	1.4	5	1.7	4	0.6571
	1.5	5	1.7	4	0.6577
	1.6	5	1.7	4	0.6584
	1.7	5	1.8	4	0.6590
0.50	0.1	1	0.1	1	0.6828
	0.2	1	0.2	1	0.7328
	0.3	2	0.3	2	0.7802
	0.4	2	0.4	2	0.8002
	0.5	2	0.5	2	0.8202
	0.6	2	0.6	2	0.8402
	0.9	3	0.9	3	0.8900
	1.0	3	1.0	3	0.8983
	1.5	4	1.5	3	0.9399
	1.6	5	2.0	4	0.9575
	1.7	5	2.0	4	0.9592
0.70	0.1	1	0.1	1	1.2001
	0.2	1	0.2	1	1.2701
	0.4	2	0.4	2	1.391
	0.5	2	0.5	2	1.4315
	0.6	2	0.6	2	1.472
	1.0	3	1.0	3	1.6252
	1.1	3	1.1	3	1.6475
	1.4	4	1.6	3	1.7583
0.80	0.1	1	0.1	1	1.6379
	0.4	2	0.4	2	1.8652
	0.5	2	0.5	2	1.9204
	0.7	3	0.8	2	2.0858
	1.0	3	1	3	2.1909
0.90	N/A	N/A	N/A	N/A	N/A
0.95	N/A	N/A	N/A	N/A	N/A

Thus, for certain given "actual" marginal costs of sampling c_s, we have identified corresponding "imposed" marginal costs of sampling $\widehat{c_s}$ such that the social optimum for c_s coincides with the Nash equilibrium for $\widehat{c_s}$. The difference $\widehat{c_s} - c_s$ can be viewed as a tax (or subsidy) that a system operator levies on individuals so that when each customer behaves selfishly to minimize her cost, the system converges to the social optimum corresponding to the marginal sampling cost c_s.

These results also indicate that in particular cases, the socially optimal strategy is already a Nash equilibrium (without any manipulations on sampling costs). In other words, $\widehat{c_s} = c_s$. The fact that the social optimum strategy's equilibrium is also a Nash equilibrium implies that no intervention is necessary for the system to reach the optimal steady-state distribution. This is very interesting. Note, however, that for high arrival rates ($\lambda \geq 0.90$), the social optimum never coincides with a Nash equilibrium. Additionally, note that we only considered sampling costs in the interval $[0, 2]$ in increments of 0.10.

Given a cost of sampling and assuming all other customers are following the socially optimal strategy, Figures 9 and 10 show how the Nash equilibrium strategy for player i compares to the socially optimal strategy. In other words, given c_s and assuming $T_{-i} = T^*$, the following figures show $(T_i^{\text{NE}} - T^*)$. If $T_i^{\text{NE}} \neq T^*$, we refer to this as a deviation.

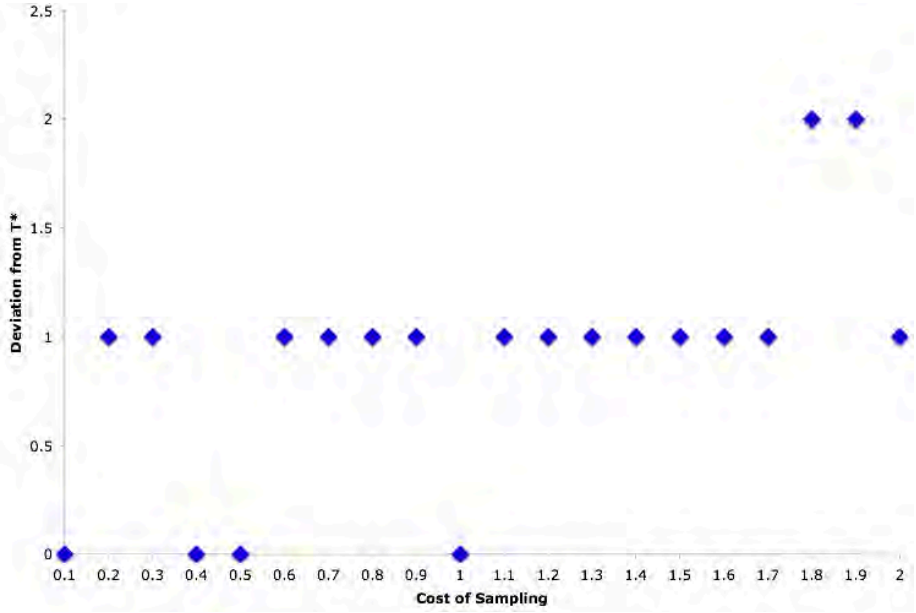


Figure 9: Comparison of T_i^{NE} to T^* for $\lambda = 0.80$

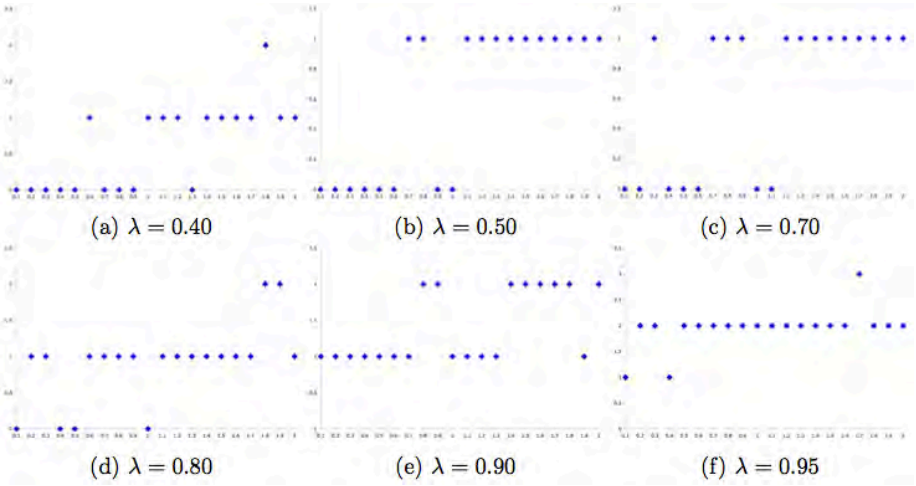


Figure 10: General Comparison of T_i^{NE} to T^*

Further work can be done to solve for all Nash equilibria of this game for all sampling costs. Additionally, further work can be done to explore uniqueness of these Nash equilibria.

5.3 Prior Work

5.3.1 Basic Game - The Supermarket Game

In a paper titled "The Supermarket Game," Xu and Hajek [16] analyze a classic supermarket game. The classification of "game" stems from the facts that the waiting time of a customer depends on other customers' choices and that often customers attempt to find the shortest queues to join. The basic model remains the same: customers arrive according to a Poisson process, N processors serve customers with a FIFO discipline with exponentially distributed service times, and ties are broken uniformly at random. Xu and Hajek add costs to the model. Each customer faces both a cost for waiting time and for sampling. Customers are assumed to be selfish and aim to minimize their own total cost. However, the duo also analyzes the case of the social optimum, in which the total cost of **all** customers is minimized. Xu and Hajek seek to determine what is the optimal number of queues to sample (when a customer is selfish) and whether some customers sampling more queues is advantageous or harmful to other customers. The latter question, regarding the externality of sampling more queues, is not immediately obvious. It is true that sampling more queues will lead to a well-balanced system and therefore reduce the average waiting time; however, it also prevents future customers from finding less-loaded queues.

The pair first examines the case with homogeneous waiting cost. They arrive at the following conclusions:

1. There always exists a mixed strategy Nash equilibrium for $\lambda < 1$.
2. The Nash equilibrium is unique for $\lambda \leq \frac{1}{\sqrt{2}}$.
3. The Nash equilibrium is unique if and only if a local monotonicity condition is satisfied.
4. The sampling of more queues by some customers has a positive externality on other customers in the mean field model. This implies that no more queues are sampled for any Nash equilibrium than for the social optimum. However, the sampling of more queues can have a negative externality in the case of finite N servers.
5. Multiple Nash equilibria exist for a particular example with $\lambda = 0.999$.
6. The Nash equilibria for $\lambda = 0.999$ is unique if customers are limited to being able to sample only one or two queues.

They then went on to study the case with heterogeneous waiting times. Specifically, they utilized a non-degenerate continuous probability distribution function to determine waiting cost. A proof of the existence of a pure strategy Nash equilibria was achieved.

In conclusion, a Nash equilibrium always exits when $\lambda < 1$ and is unique for $\lambda \leq \frac{1}{\sqrt{2}}$ with homogenous waiting costs. The idea of a supermarket game is truly intriguing. There is very little literature on this topic currently, leaving much room for expansion. How

do these results change with general service times or heterogeneous service times? Is it possible to provide an incentive that will move the Nash equilibria closer to the social optimum without losing throughput or hurting efficiency? What happens if asymmetry is introduced to the customer strategy? Many interesting topics remain to be addressed in this field.

5.3.2 Extensions of the Basic Game: Queue-Length-Based Scheduling

Manjrekar, Ramaswamy, and Shakkottai [17] examine the scheduling game of smart phone applications competing for service from base stations. They seek to find a scheduling algorithm, through auctions in which the players compete for service, that will replicate the advantageous results of the Longest-Queue-First (LQF) policy. As one might think, the LQF policy consists of each server servicing the longest of the queues requesting service. The major benefit of such a policy is minimizing the expected time of the longest queue in the system.

To try and replicate the LQF policy's results, Manjrekar et al. analyze a second-price auction. In such a regime, the highest bid wins and is then charged the rate of the second-highest bid. Such an auction is proven to promote truth in bids (or preventing lies aimed at "beating the system"). In the proposed model, smart phone applications send service requests, experience a waiting cost, and bid for service from its base station. In order to generate a bid, each player models opponents through a distribution over the possible action spaces. The player then chooses a best response. A mean-field equilibrium is reached if this best response can be found in the distribution. Manjrekar et al. seek to find an equilibrium in the auction that will replicate the beneficial results of the LQF policy.

Through a mean-field analysis of this scheduling protocol in cellular networks and verification through simple simulations, Manjrekar et al. discover that as the number of servers approaches infinity, performing a second-price auction at each server does in fact perform as well as a queue-length-based scheduling algorithm. A natural extension of this work would be to add classifications of different types of applications. In practice, requests from different applications have different cost functions and arrival rates. This simple modification of the model could add great insight into this area of study.

6 Conclusions and Further Work

We have shown that when a cost is assigned to polling in a load balancing problem, implementing a threshold to dynamically determine the number of queues to sample significantly reduces the expected total cost of customer i. For all λ and sampling cost greater than zero, the threshold model always performs better than the corresponding case without a threshold. The optimal thresholds found, for a given λ and cost structure, coincide with the system's social optimum. For certain given "actual" marginal costs of sampling c_s, there do exist corresponding "imposed" marginal costs of sampling $\widehat{c_s}$ such that the social

optimum for c_s coincides with the Nash equilibrium for $\widehat{c_s}$. This implies that a tax (or subsidy), calculated from the difference $\widehat{c_s} - c_s$, can be levied on individuals such that the social optimum for c_s coincides with the Nash equilibrium for $\widehat{c_s}$. Because sampling does realistically cost the system, we believe this model will prove useful in the design of many load balancing systems.

Still, several open questions remain. The most obvious questions seek to discover analytic solutions to the cost optimization and pure strategy Nash equilibrium problems. The most interesting expand upon the described threshold supermarket game. How do the results change when mixed strategies are considered? Would incentives truly push the system into the socially optimal equilibrium in practice? Many implications of our threshold model and game remain to be explored.

References

[1] Vvedenskaya, Nikita, Dobrushin, Roland, and Karpelevich, Fridrikh. "Queueing System with Selection of the Shortest of Two Queues: An Asymptotic Approach." *Problems of Information Transmission*, 32:15-27, 1996.

[2] Mitzenmacher, Michael. "The Power of Two Choices in Randomized Load Balancing." *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1094-1104, 2001.

[3] Farias, Vivek, Moallemi, Ciamac, and Prabhakar, Balaji. "Load Balancing with Migration Penalties." *In Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2005.

[4] Squiillante, Mark and Lazowska, Edward. "Using Processor-Cache Affinity Information in Shared-Memory Multiprocessor Scheduling." *IEEE Transactions on Parallel and Distributed Systems*, 4(2):131-143, 1993.

[5] Vöcking, Bethold. "How Asymmetry Helps Load Balancing." *Journal of the ACM*, 50(4):568-589, 2003.

[6] Azar, Yossi, Broder, Andrei, Karlin, Anna, and Upfal, Eli. "Balanced Allocations." *In Proceedings of the 26th ACM Symposium on Theory of Computing*, 593-602, 1994.

[7] Mitzenmacher, Michael, Prabhakar, Balaji, and Shah, Devavrat. "Load Balancing with Memory." *In Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[8] Prabhakar, Balaji, and Shah, Devavrat. "The Use of Memory in Randomized Load Balancing." *In Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2002.

[9] Mitzenmacher, Michael. "Analyses of Load Stealing Models Based on Families of Differential Equations." *Theory of Computer Systems*, 34:77-98, 2001.

[10] Minnebo, Wouter and Van Houdt, Benny. "A Fair Comparison of Pull and Push Strategies in Large Distributed Networks." *IEEE/ACM Transactions on Networking*, 2013.

[11] Tsitsiklis, John and Xu, Kuang. "On the Power of (Even a Little) Resource Pooling." *Stochastic Systems*, 2:1-66, 2012.

[12] Breitgand, David, Cohen, Rami, and Raz, Danny. "On Cost-Aware Monitoring for Self-Adaptive Load Sharing." *IEEE Journal on Selected Areas in Communications*, 28(1): 70-83, 2010.

[13] Bramson, Maury, Lu, Yi, and Prabhakar, Balaji. "Randomized Load Balancing with General Service Time Distributions." *In Proceedings of the ACM Special Interest Group on Computer Systems Performance*, 2010.

[14] Bramson, Maury, Lu, Yi, and Prabhakar, Balaji. "Decay of Tails at Equilibrium for FIFO Join the Shortest Queue Networks." *The Annals of Applied Probability*, 23(5): 1841-1878, 2013.

[15] "Randomized Routing Schemes for Large Processor Sharing Systems with Multiple Service Rates."

[16] Xu, Jiaming and Hajek, Bruce. "The Supermarket Game." *Stochastic Systems*, arXiv: 1202.2089v2, 2012.

[17] Manjrekar, Mayank, Ramaswamy, Vinod, and Shakkottai, Srinivas. "A Mean Field Game Approach to Scheduling in Cellular Systems." *Stochastic Systems*, arXiv: 1309.1220v1, 2013.